FINAL REPORT

"Design Methodology and Strategies Investigation for Complex Integrated Naval Systems"

Contract #: N00014-04-1-0127

SUBMITTED TO:
Mr. Anthony J. Seman, Office of Naval Research
(email: Anthony_Seman@onr.navy.mil)

Office of Naval Research
875 North Randoph Street
Arlington, VA 22203-1995

SUBMITTED BY:
Georgia Institute of Technology
School of Aerospace Engineering
Aerospace Systems Design Laboratory (ASDL)
Atlanta, Georgia 30332-0150

Contract Start Date: November 17, 2003
Contract End Date: November 16, 2006

PERIOD OF PERFORMANCE:
November 17, 2003 to November 16, 2006

February 14, 2007

Principal Investigator:
Professor Dimitri N. Mavris
Director
Aerospace Systems Design Laboratory
School of Aerospace Engineering
Georgia Institute of Technology
Phone: (404) 894-1557
dimitri.mavris@ae.gatech.edu

1 Introduction

As the mission and performance demands for naval ships have increased, they have become more complex comprising an increasing number of heterogeneous interdependent subsystems. This increased complexity requires new methods for the design and operation of these naval systems. The Georgia Institute of Technology Aerospace Systems Design Laboratory (ASDL) is helping the Navy change its design practices to achieve reduced total ownership costs, increased survivability, and increased mission effectiveness through an initiative called Integrated Reconfigurable Intelligent Systems (IRIS). Using traditional systems engineering practices for the early design process followed by an integrated design environment, IRIS seeks to shift ship design to a distributed intelligent control architecture through increased automation.

The Integrated Engineering Plant (IEP) is an example of the kind of technological innovation that will contribute to the Navy' capability of achieving its future goals. The following report details the research accomplished by ASDL in developing and applying the IRIS concept to the IEP platform. Section 2 of the report lays the foundation for subsequent sections by presenting the background and motivation for developing an IRIS based IEP concept for naval vessels. Section 3 discusses more fully the components and characteristics that describe the IRIS concept as applied to naval surface combatants. Although IRIS is being applied specifically to the IEP in this research, the IRIS concept is fully generic in nature and can be applied to any complex dynamical system.

2 Background

During the last decades, incremental improvements in ship design, operation, and capability have been achieved. Today's conventional ships have independent shipboard propulsion and electrical power systems with centralized systems associated with the rest of the shipboard engineering plant and machinery infrastructure. Almost all Navy ships still use human-in-the-loop communications and decision-making techniques, which cause increased ownership cost. Figure 1 shows the need for shipboard automation as personnel costs, shown in red, represent more than half of the Navy's budget for a given year.

Experience also shows that there is also the need for increased survivability. Modern ships comprise many interdependent subsystems. These systems must be robust in their ability to survive extreme events that may damage or disrupt vital services to parts of these systems while being capable of multiple types of missions. As a result of their inherent complexity, these systems will exhibit emergent behaviors, which can be difficult to predict. Therefore, methods must be developed to try and ensure that these systems will exhibit graceful degradation of performance when subjected to attack or unanticipated events.

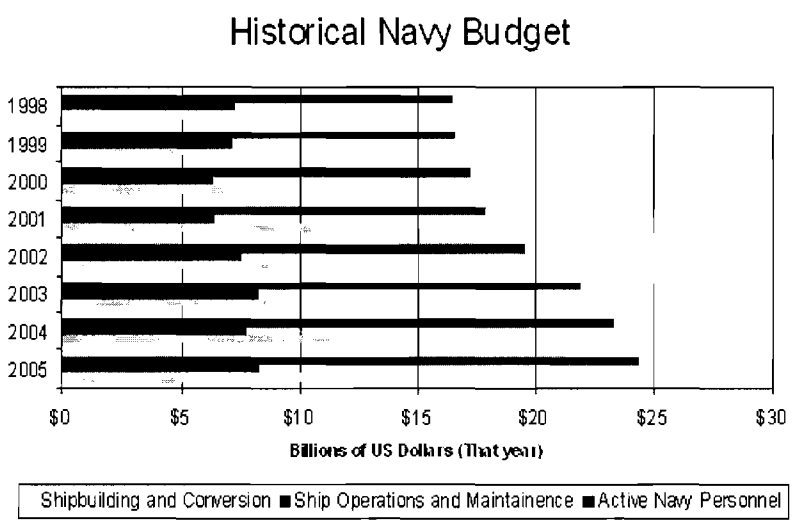


Figure 1: Historical dominance of personnel cost in Navy budget [Congressional Budget Data, 2005]

Therefore, with the rapidly changing fiscal and threat environment modern ship design is putting more emphasis on reducing operating cost and manning workload, and increasing ship survivability and mission effectiveness.

2.1 DDG Class and Smart Ship

Many efforts have been made to meet such naval requirements. With the installation of a series of revolutionary automation and survivable systems, the Arleigh Burke class is considered to be the most advanced and survivable ship of the fleet. Nonetheless, the pace at which technology is advancing makes these systems be outdated by today's standards. The future surface combatant will have to provide even higher levels of survivability and dependability while reducing the operating cost. The next-generation multi-mission destroyer, DDG-1000, has as a prime objective to drastically reduce manning. The first flight is envisioned to be operated by 150 servicemen and women and officers. This is more than 50% reduction when compared to DDG-51 and subsequent Arleigh Burke class destroyers. Future flights for the DDG-1000 aim to require only 95 operators, including officers, support crew and flight deck crew. This reduction in manning demands an unprecedented level of automation, both in terms of nominal operation and damage control. In addition, as depicted in Figure 2, the other significant changes between current platforms such as the DDG 51 and the future naval surface combatants like the DDG 1000, is the transition to the Integrated Power System for the purpose of optimizing the manning.

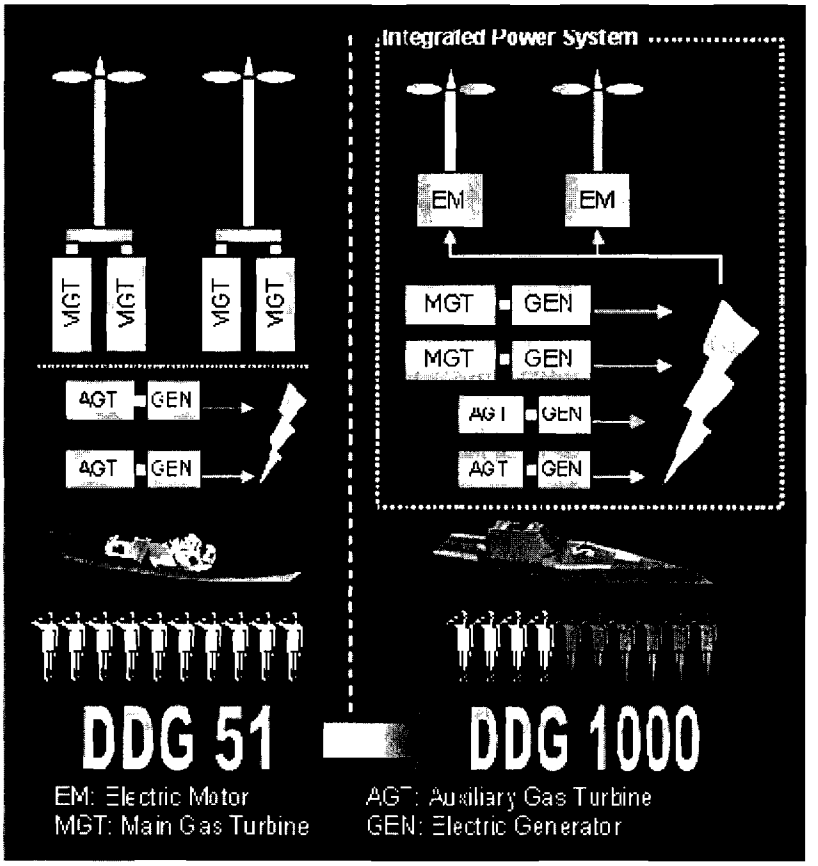


Figure 2: Transition from Arleigh Burke class to the new Zumwalt class

In order to investigate the capability to automate its assets, the US Navy and Coast Guard have undergone some testing of their "Smart Ship" research. The USS Yorktown (CG-48) was modified and automated to reduce the workload, manpower requirements and cost while enhancing the combat readiness and quality of life of the crew. The Yorktown incorporated damage control and engineering systems which automated many of the routine daily tasks, was operated with integrated bridge, effectively reducing the repetitive tasks that the crew had to perform this allowed them to concentrate on their war fighting. The Smart Ship program is a demonstrator of what a small amount of automation can produce in terms of added capabilities and reduced costs.

2.2 The Integrated Engineering Plant

To meet the requirements of the next generation surface combatant, the Office of Naval Research (ONR) proposed an Integrated Engineering Plant (IEP) concept. IEP is a unified system that removes traditional system-level barriers between the various ship plants, such as propulsion, weapon, electrical and cooling systems. Thus, the ship plants can share the resources and information managements systems which leverage the resources and deliver the information to the plants from a system point of view. IEP is a highly decentralized system in which plant components can perform the predefined or self controlled

tasks, as shown in Figure 3. In addition, the IEP system allows the next generation Navy ships to operate under major disruptions involving cascading failures and provide continuous mobility, power, thermal management and fluid transfer for vital shipboard systems, as a result, reducing manpower requirements and increasing overall ship survivability and mission effectiveness.

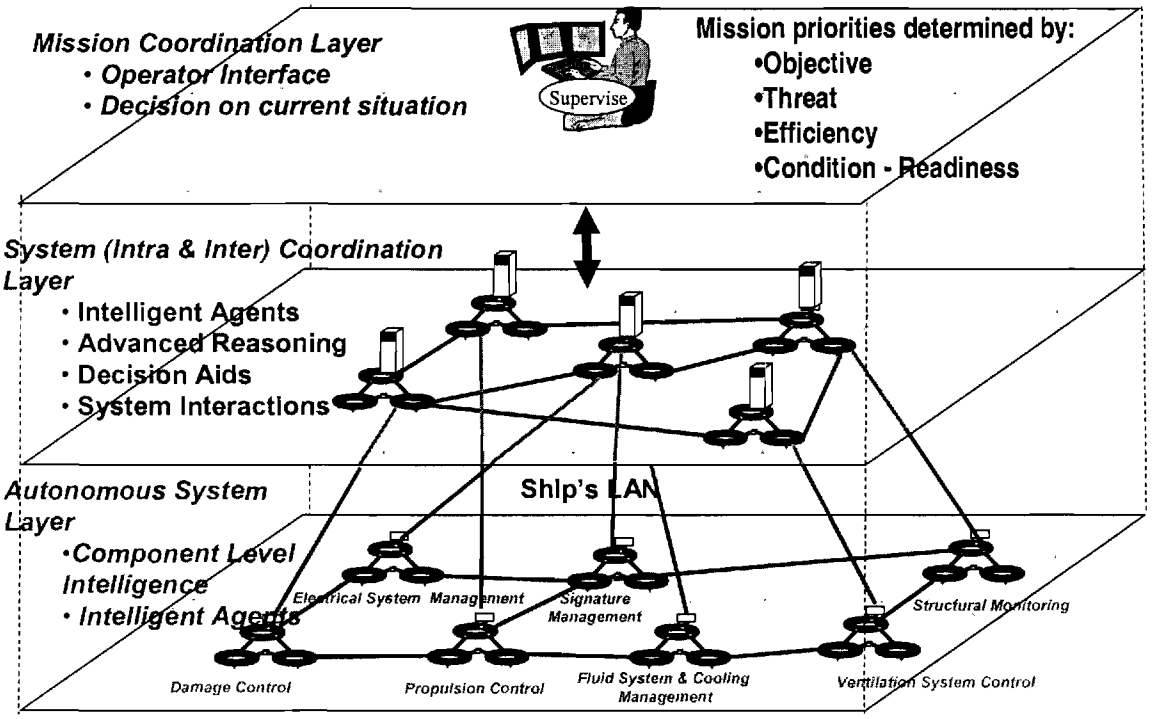


Figure 3: IEP Concept [Walks and Mearman, 2005]

The IEP attempts to integrate the ships engineering systems and resource allocation systems, including the electrical systems, propulsion systems and auxiliary systems. With the integrated system, automatic reconfiguration can be enabled for normal, casualty and damage scenarios. This will allow for the increased survivability of the ship as well as the resource management. Reliability, survivability and fight-through capability can all be achieved with the IEP enabled technology. Furthermore, by implementing a modular design architecture and allowing more optimal maintenance procedures, the total ownership cost will be reduced.

As stated previously, IEP will integrate many systems onboard the ship. The near term goal is to integrate models of distributed fluid, thermal and power systems, with shipwide autonomous sensor and data networks." This integration will leverage previous research and development, such as the Integrated Power System (IPS) concept (1996-2001, UT-Austin, Purdue, USC), The Reduced Ship Crew by Virtual Presence (RSVP) technology improvements (1999-2001, Draper Labs) and the Damage Control –

Automation to Reduce Manning (DC-ARMS) technology improvements (2001, Naval Research Laboratory, Navy Technology Center).

The objectives of the IEP implementation include reduced ownership costs, increased survivability and increased mission effectiveness. One way to reduce the total ownership costs of the fleet through the IEP is to use automation to optimize the manning required on ships. Therefore, an advance approach should be identified to enable the ship operates autonomously. On the other hand, in order to increase the ship survivability and mission effectiveness, a ship in battle must be able to instantaneously communicate with other assets in the theatre. In addition, a task force of ships needs the ability to coordinate attacks on multiple targets and dynamically update plans as the battle progresses. To fulfill these objectives, an organized, informed method must be developed to, with all interdependent subsystems working together, achieve an effective, efficient mission.

3 The Integrated Reconfigurable Intelligent Systems Initiative

The solution to the IEP problem proposed by ASDL has been christened Integrated Reconfigurable Intelligent Systems (IRIS). The IRIS initiative was created as a solution to the request for a system that addresses the areas of improvement of the conventional ship as well as meets the new requirements. Having contributed on this effort, ASDL has taken part in the project in the development of a robust design environment and design methodologies for future naval surface combatant systems sponsored by the Office of Naval Research (ONR) [ASDL, 2004]. The focus of the IRIS initiative is to create a M&S environment for the IEP as a part of the entire ship system-of-systems. This state-of-the-art on-board intelligent computer system encompasses all electrical, mechanical and damage control systems. The IRIS project will enable ships to be designed to meet the criteria and be optimized for the primary missions utilizing the most suitable technologies for the specified missions.

ASDL has developed a toolbox of methods which are combined to create processes for addressing problems of varying complexity. Although the IRIS concept is being developed to address IEP, the results of this research will be applicable to complex systems of systems problems in other areas. ASDL's method for IEP will include concurrent examination of all projects and their enabling technologies, selection of the most robust combination of all possible technologies and alternative solutions and the development of an integrated M&S environment. With the ASDL solution, the IEP M&S environment will allow for the design of distributed component level intelligence, decision aids, fault tolerant information networks, automated machinery health management and reconfigurable power and auxiliary systems.

The systems that will be addressed by IRIS are characterized by their high degree of interdependence and heterogeneity, their ability to adapt to evolving conditions and their need to extensively collaborate

with other systems. IRIS for the IEP is a design methodology utilizing the simulation tools created in the ship community. The simulation tools can be replaced with those of the aeronautical, automotive or space industry and the IRIS methodology can be applied to the design and monitoring of dynamic interdependent systems.

The first "*I*" stands for *integrated design*. The methods developed by the IRIS initiative will allow for the integration of multi-level heterogeneous systems. The design of the system is shaped by the integration of intelligent and reconfigurable subsystems, which help to reduce manpower requirements and increase mission effectiveness, survivability, and reliability of the overall system.

The "*R*" stands for *reconfigurable operations*. IRIS will allow the system to diagnose the current state of the ship and reconfigure the resources onboard the ship by either autonomously reconfiguring the ship or the system will suggest solutions to a human decision-maker to deal with the current situation. Ships will be designed using a modular architecture that will allow for use of similar equipment and have a lower development cost which will allow the ship to remove and replace particular platforms seamlessly as new technologies become available.

The second "*I*" stands for *intelligent components*, meaning autonomous systems. With the implementation of sensors and smart sensor technologies in combination with advanced networking solutions, the ship can accurately sense and assess situations and suggest solutions. The system is aware of its surroundings through the gathering of data from sensors onboard the vehicle and then either makes decisions autonomously or defers the decision to a human operator.

Finally, the "*S*" stands for *system-of-systems*. This is an integrated system solution. With the advancing of enabling technologies the IRIS solution will implement the various technology solutions to design an IEP enabled ship including people, products and processes that provide a capability to satisfy a stated need or objective. Theories of system design such as multidisciplinary design, parametric design and optimization will be included in the IRIS M&S environment. The IRIS methodology will integrate the design of complex systems such as the propulsion system, navigation, weapons, radar, fire detection, fluid system and damage control, network-centric operations, and auxiliary subsystems into a single system where the subsystems communicate amongst themselves to coordinate their activities.

An IRIS designed ship will be self-monitoring, self-assessing, self-reacting and efficient because of the IEP initiatives and technologies. Self-monitoring will enable continuous sensing of all ship-related operations and encourage a system that is knowledgeable of both current and impending failures. The IRIS ship will be able to acquire inputs from all subsystems and sensors and either automatically diagnose and act on the best course of action or pass the information to a decision-maker. A critical aspect of the assessment function is the ability to infer when data is unavailable or incorrect. The self-reacting feature

will allow the resources and commands to be distributed as necessary and prepare to compensate and continue to function in the event of system failures. This could include human supervision to ensure the correct decisions are made for the given scenario. These three features as well as the incorporation of technology to optimize manning will improve the effectiveness of the ship.

New emerging technologies such as advances in communications, sensor technology, intelligent algorithms and modeling can enable the success of the IRIS initiative and propel the design of systems like the IEP into the future. The electronics have become more reliable, smaller, lighter weight, cheaper, efficient and robust. These advances in electronics have enabled the implementation of commercial-off-the-shelf (COTS) technology onboard a ship. The IRIS environment will allow for increased knowledge and robust solutions early in the design stages, which will affect the end cost of the project by reducing the number of changes required in the final design phases and manufacturing.

There are three essential functions that characterize every IRIS system. These are sensing, assessing and reacting, an iterative process as represented below in Figure 4.

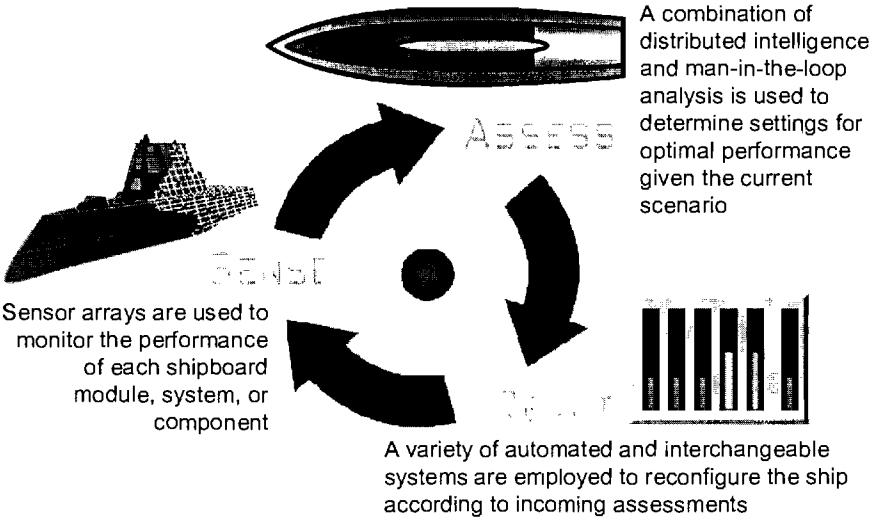


Figure 4: The three IRIS functions

The ability to sense changes in the state onboard a ship and its surroundings will be vital for automating operations. This will be described in depth and describe how state of the art technology will be implemented in the design environment to accomplish the goals listed above. Using the sensor data, an IRIS-designed ship will have the ability to automatically analyze and assess the data. The assessment mode will use the data from the sensors and identify the methods for avoiding or mitigating any foreseen problems. Technology will be in place to offer methods of damage mitigation and initiate a physical system reconfiguration to minimize the effects of the change in the state of the system. This integrated system and technologies onboard an IRIS designed ship will have the ability to automatically sense,

assess and react to changes in constantly changing scenarios. The next three subsections will describe the three IRIS functions in more detail.

3.1 Sense Function

An IRIS-designed ship will contain a network of system sensors capable of assessing the state of the system. This network of sensors will track ship motion, monitor personnel status, detect damage, monitor machinery health, and evaluate weapon system readiness. Each ship subsystem will contain sensors to monitor its respective health status.

One such network of sensors is the Intelligent Machinery Health Monitoring (IMHM) that monitors the mechanical and thermodynamic health of turbo-machines [Dunnington et al., 2003]. With sensors already in place, the IMHM can be implemented on a component level with the monitoring and control coming from the network-based sensors. Sensors will provide the ability to gather details about the health of machinery and subtle changes that could cause future problems in operations. The use of advanced computing to predict failures along with continuous system health monitoring will allow for immediate corrective action to be taken in the event of system failure. Delta Airlines experienced very promising results after contracting SmartSignal to implement a similar system to help do preemptive maintenance and reduced operating costs [Anderson, 2003]. They have significantly decreased unplanned maintenance and have successfully transitioned to scheduled/optimal maintenance.

The fluid systems and damage control system onboard ships are critical systems that can also utilize the advanced network of sensors [Dunnington et al, 2003]. These fluid systems include fire mains, seawater, fresh and chilled water systems, fuel systems and air systems; these systems are key to enhancing survivability. Autonomous monitoring and control of these systems is needed to enable reconfigurability and fast response time. The emphasis in this area will be on the fire suppression system because of the grave threat posed by fires onboard a ship [Gillis et al., 2003]. Automated control of valves and pumps within this system will enhance the capability to combat fires by isolating the damage and reconfiguring the available resources for optimum continuous operation.

Another application of the network-based array of sensors is the Reduced Ship Crew through Virtual Presence (RSVP) [Seman, 2001]. With the use of the onboard sensors, RSVP can reduce manning requirements and operating costs by the installation of an intracompartmental sensor array that monitors four functional areas - environmental, structural, personnel and machinery. These network sensors will detect fires, flooding, machinery faults and failures, personnel status and damage in real time. Other sensors, called Personnel Status Monitors (PSMs), can be used to monitor crew health status remotely during times of crisis and alert the proper medical personnel. A single watch-station would allow one operator to monitor all of these systems remotely using sensor information and video monitors to decide

the best possible course of action. The challenge in creating a sensor system like the RSVP is optimizing the placement and type of the sensors, as well as the modeling of the sensors in the design environment.

3.2 Assess Function

The next step in designing an IRIS ship is the ability to assess the information provided by the network of sensors. The assessment will then incorporate distributed intelligence and man-in-the-loop analysis. Given a specific mode of operation, the sensor information will be used to determine the optimum alignment of the ship given the subsystem health information and any predicted failures in the near future.

The assessment capabilities of the ship will utilize a distributed intelligence system to reduce the dependency on human operators, which will also decrease cost and increase efficiency [Dunnington et al., 2003]. The automation software for this intelligence system must be capable of making unsupervised decisions within a closed loop, the complexity of which can be greatly reduced by segregating the architecture into independent units. Although each computing node would have minimal responsibility on its own, robustness would depend on either a redundant architecture or data fusion.

The success of a system of this advanced nature depends on its reliability and survivability in all ship operating conditions. It requires component level automation under a hierarchy of systems with distributed intelligence nodes connected over a communications network. The key to this is the ability to isolate the hardware and software from remote failures, with the capability to reroute through redundant paths to avoid cascading failures.

The most challenging scenario for the survivability of a distributed intelligence system is a wartime scenario. The autonomy and integration of all systems can revolutionize both day-to-day and wartime ship operations. To take full advantage of these new technologies, the ship's combat information center will implement the integrated autonomous system, enabling the user to operate all of the attack systems in a very functional, responsive manner using the integrated controls [Ulrich and Edwards, 2003]. The operator of a watch station will directly interact with a visual, top-level interface with on-demand access to additional information and procedures as needed [ONR, 2004]. A system with these assessment capabilities will present a robust solution for all system operators, offering every possible bit of information that could be required in a given scenario. Current plans include using a secure login system that only allows personalized access and control.

The critical step in the ability to assess the situation and develop a plan of action that will result in the most beneficial end-state for the ship is in understanding the information provided. The quality of the data transmitted by the sensors will be compromised by damage and wear-and-tear of the system. When the data are neither accurate nor available, the system has to be able to infer what the state is. The inference will have to be probabilistic by nature and one of the most promising solution involves the use of

Bayesian Networks to estimate the likelihood of each state for a given set of evidences (sensed data) [Bajwa and Sweet, 2003].

3.3 React Function

To complete the design using the IRIS methodology, the ability of the system to react must be incorporated. This is the functionality that takes advantage of the automated distributed intelligence system to physically reconfigure the different ship modules based on the sensors and the assessment of the information [ASDL, 2004]. The systems that will benefit from automation include the machinery control systems, damage control reporting, integrated bridge system, integrated control system, wireless communications, automated condition assessment, ship surveillance, collision avoidance, and pre-hit system reconfiguration.

In order to automate the next generation of warships in a cost and time effective manner, the necessary automation technology must come from commercial automation and network communications technologies. These COTS systems must be able to withstand the harsh environment of wartime operations onboard the ship to ensure no loss of functionality when it is needed the most. Even considering the potential added effort to make COTS seaworthy, the COTS systems are readily available, inexpensive, and scalable compared to the development of a completely new system. Possible new technologies to be included in such a system are wireless sensors, knowledge projection, Power Electronic Building Blocks (PEBBs) [Ericsen, 2006] and power reconfiguration technologies.

The IEP is the initial platform for accomplishing what is required for the IRIS project. The IEP strategy leverages Integrated Power Systems (IPS) development and translates the benefits of increased survivability and reconfigurability to the ship's engineering plant and infrastructure. IEP integrates propulsion, electrical and auxiliary system architecture and resource management systems to enable automatic reconfiguration and control of mission critical resources under normal operation and damage conditions. This automated reaction system will satisfy the needs discussed previously for increased ship reliability, survivability and fight-through capability.

Even with the aid of the IEP project advances, there exists a gap in the ability to implement the assess function of IRIS. IRIS will leverage IEP information in the creation of the overall design environment simulation tool. The research at ASDL will address the integration of existing ship subsystem modeling codes into a single environment and create objective functions, which can be used to evaluate the dynamic performance of a variety of designs under multiple scenarios. An IRIS design will determine the necessary action to take after an initial sensing and assessment phase. Other disciplines such as aerospace design will also be explored for sensor and reconfiguration technologies that may be transferred to ship design.

4 Motivation

This revolutionary change in naval architecture and ship engineering requires a total ship systems engineering design approach which is capable of formulating design methods and implementing tools on ship systems. Eventually, a framework can be developed to provide sophisticated naval systems that are capable of making intelligent, extensive and autonomous decisions based on situational awareness for the purpose of maximizing mission effectiveness.

The Aerospace Systems Design Laboratory (ASDL) at the Georgia Institute of Technology has proposed the Integrated Reconfigurable Intelligent System (IRIS) framework as a possible solution to the IEP concept. The IRIS integrates many intelligent systems onboard to collect the information about the environment and ship state, assess the situation and then determine a best course of action to take in order to reconfigure the ship into the state that most suitable to handle the situation at hand. Therefore, the IRIS designed ship is envisioned to be self-monitoring, self-assessing and self-reacting.

The successful design of IRIS should encapsulate all characteristics of complex hierarchical systems and several issues associated to them must be addressed during the design process. These complex systems often operate in a hostile environment in which the information is usually uncertain and frequently changes over time. As a result, new design methods need to be developed to consider the dynamic characteristics of the system and allow for the uncertainty regarding the system's operations, so that a robust solution can be obtained to increase the ship's mission effectiveness.

The US Navy's pursuit of more affordable, efficient and survivable platforms leads to extensive integration requirements of heterogeneous subsystems. In order to find the optimum or robust solution of a ship design for optimal mission effectiveness, the integrated subsystems need to be well studied and understood. Therefore, an environment is needed, for the purpose of integrating multiple physics-based models to accurately simulate the dynamic behavior that the system exhibits. To meet the mission effectiveness and ship survivability objectives, a distributed intelligent control architecture should be developed for the implementation of the autonomous decision making process. This process will be strongly aided by the creation of a modeling and simulation environment to represent the total operations of typical naval ship architecture. In addition, a human-in-loop study should be performed to investigate how the system prioritizes its tasks, how it interacts with the human operators and how any unsupervised operation can be avoided.

In order to find the optimum or robust solution to the system design and operation, a large number of configurations need to be evaluated. Therefore, an approach is needed for the rapid assessment of the alternative configurations for accelerating the design selection process.

The challenge of designing next-generation ship systems that meet operational goals for system mission effectiveness, environmental compatibility, and reduced cost has grown to the point that traditional design methodologies are becoming ineffective. This situation is definitely supported by process related obstacles, such as demanding analysis requirements for complex system, large number of objectives and constraints to be evaluated, and the multitudes of uncertainty sources that appear in current design problems.

Furthermore, there is no standard and systematic method for integrating, validating, verifying subsystem models of complex systems. Complexity in this context entails nonlinear interdependencies, distributed control requirements, combination of discrete and continuous parameters and an ever evolving emergent behavior. All these characteristics lead to the need to study the time-dependent behaviors of the integrated system. Eventually, this will assist in formulating a suitable and efficient integration strategy that will allow for a smooth and seamless integration scheme of the different subsystem models.

4.1 Improve Situational Awareness

Situational Awareness is the main enabler for improving a naval system's mission effectiveness. In other words situational awareness can be defined as the ability of collecting and assessing information associated with the system and its surroundings. This information is necessary for determining the overall operational status of the system at a given time instant, creating alternative plans of action and evaluating the alternative plans to find the most suitable action that the control system should adopt for achieving the prerequisite level of mission effectiveness.

4.2 Improve Mission Effectiveness

Mission effectiveness is indicative of the system's total capability of achieving its current mission's objectives and includes how well can those objectives be achieved. Three main contributions to mission effectiveness can be considered and these are survivability, reconfigurability and system reliability. Survivability is associated to a concept which includes all aspects of protecting personnel, weapons, and supplies while simultaneously deceiving the enemy. The Department of Defense defines survivability as tactics that include building a good defense; employing frequent movement; using concealment, deception, and camouflage; and constructing fighting and protective positions for both individuals and equipment. Reconfigurability is the ability of the system, not only to withstand the consequences of any event that can affect its operations, but also to be capable of returning back to an operational status after the event has occurred. Reconfigurability improves survivability. System reliability is the inherent probability of failure associated with the system. Therefore survivability, reconfigurability and system reliability are the three properties of the system that are responsible for improving mission effectiveness.

5 Methodology

Improved mission effectiveness and enhanced survivability characteristics are one of the main objectives of the IEP [Lively et al., 2005]. The IRIS framework as a proposed solution to the IEP should involve a set of processes that will ensure that the proposed IEP implementation will satisfy the mission effectiveness requirements that are set by the Navy for the next generation of Naval ship combatants. Design for mission effectiveness therefore must be ensured by the application of a design methodology that seeks to improve the system design based on objective function responses associated to the survivability, reconfigurability and reliability of the system.

Design for mission effectiveness is coming to supersede previous design methodologies that have been developed in ASDL, such as the Design for Affordability initiative funded by ONR. The Technology Identification Evaluation and Selection (TIES) [Kirby, 2001] methodology is one of the most popular design frameworks that were developed in accordance to the requirements of this initiative. It involves design space explorations that are subject to constraints of a technical and economic nature. These constraints either are oriented from the physical and societal environment, or just defined by the customer. Technical feasibility and economic viability are expected to be achieved, but it seems that most of the time this is something that does not appear automatically. In order to create some feasible or viable design space, innovative (or not) technologies should be considered, which eventually should alter the behavior of the system with respect to the imposed constraints. After such technologies are identified and explored in their entirety, it is necessary to end the process with a decision making session. This session should provide the designer with the best choice concerning the combinations of technologies that he could add to the system, in order to achieve the most favorable behavior towards the constraints.

The TIES design methodology can guarantee the technical feasibility and the economic viability of the selected design. However, there is no insight as to how this design will perform under a different set of events. A scenario event can include enemy attacks (missile or torpedo attack on a naval vessel), extreme environmental conditions (weather, climate), system failures and any other types of event that can cause s subsystem to not operate properly and reduce the overall system effectiveness. Mission effectiveness should be considered as a top level metric that describes how well can the system perform its mission and all the tasks that were assigned to it.

5.1 Design

In the context of this project, design can be characterized as an operation, generally optimization, on analysis. Analysis in turn makes use of models whose complexity has to be tailored to satisfy the analysis and depending on the complexity of the application must be made more, or less, complex. Therefore in order to design these systems models are needed, the question is then how should these models be

developed? The challenge for IRIS systems is that their time-domain performance must be analyzed, and therefore, the models should describe the behavior of the system as a function of time. Additional to the accurately portray the system's behavior over time, and because the system is composed of a large number of integrated sub-systems, which make use of different models, the IRIS model must be an integrated model, comprising of a series of sub-models that can exchange pertinent data as time advances. Therefore, design requires models, but the models' fidelity should be guided to produce the required analytical framework on which the designer can operate. Excessive amounts of detail in the models that do not provide additional benefit to the designer, are an inefficient use of resources.

5.1.1 IRIS Design and Modeling Methodology

Integrated reconfigurable intelligent systems need to be aware of their surroundings as well as themselves. The design of these systems needs to address the three IRIS functions, sense, assess and react, therefore the sensors, algorithms and actuators selection and placement needs to be studied concurrently. Figure 5 depicts the abstraction selected to model an IRIS system. The approach subdivides the steps that an IRIS system must undertake to sense, assess and react, and iterate on the process. Each of the seven steps must be modeled to capture the true behavior of an IRIS system. Over this model, different optimizations can be overlaid to ensure that the system developed is the most suitable for the tasks required.

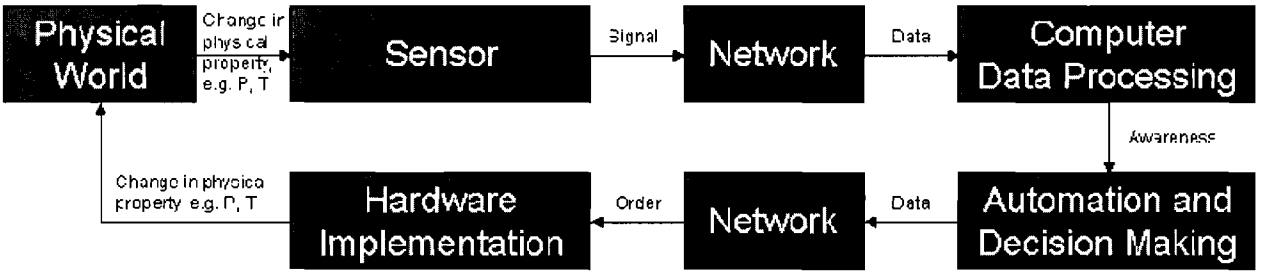


Figure 5: Process for modeling and simulation environment for sensor optimization

5.1.1.1 Sensor Optimization Methodology

The optimization of sensors will be used as an example to illustrate the method to the reader. The designer has to assess tradeoffs between different options, for example the tradeoff between having a few expensive, power demanding, bulky sensors that are highly accurate, versus many cheap, small, simple, inaccurate sensors and every combination in between. Figure 6 is a notional representation of the sensor architecture design space. The question is where should the design be? The iso-awareness accuracy lines have been notionally drawn to indicate that there is an optimum when the number of sensors and their quality is balanced. This is based on engineering intuition, but without proper knowledge of the topology of the space it is not possible to accurately perform tradeoffs. Furthermore, the designer needs to analyze

the effects of the error in the sensed information and project it to the higher levels and concurrently design the control schemes to obtain the most robust total solution. The cost for the different architectures must also be included to ensure that the design will meet affordability constraints.

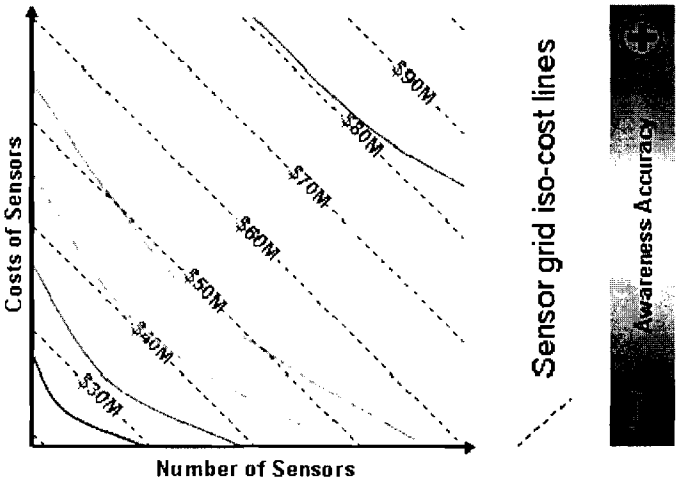


Figure 6: Balancing quality and quantity of sensors

Define the Problem

The first step in defining the problem is to determine the metrics to be sensed, e.g., pressure, temperature, structural integrity, etc. The second step is to identify metrics to be controlled, e.g., fuel flow, voltage, frequency, etc. The third step in defining the problem is identifying the metrics that reflect optimality of the system, e.g., dependability, survivability, cost, etc. If metrics are time dependent, they must be made time independent, e.g. by integrating over time, determining peak value, or other applicable operation. The final step and the more labor intensive task, is to define the baseline architecture of the sensor network.

Create the Matrix of Alternatives for Sensors

The different sensor options for each metric to be sensed can be incorporated in a matrix of alternatives as presented in Figure 7. This matrix contains up to 6 options for each state to be measured. Each option has a different set of benefits and detriments, as for example, accuracy and cost respectively. In particular, for each option, the designer should be aware of the error and data that the sensor option has. Using the matrix of alternatives allows the designer to quickly navigate the options and ensure that all possible alternatives have been considered.

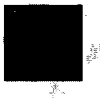
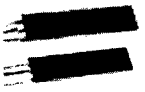
	Alternative 1	Alternative 2	Alternative 3	Alternative 4	Alternative 5	Alternative 6
Pressure	 Strain Gage	 Semiconductor	 Pirani Gauge	 Piezoelectric		
Temperature	 Thermocouple	 Thermistor	 RTD	 Bimetallic	 Semiconductor	 Radiation Thermometer
Hull Integrity	 Piezoelectric	 Strain Gage				
Smoke	 Ionizing	 Photoelectric				

Figure 7: Example Matrix of Alternatives for Sensors

Create the Sensing and Control Modeling and Simulation Environment

Once the matrix and alternatives have been identified, it is necessary to perform quantifiable tradeoffs to determine the optimum combination. The Sensing and Control Modeling and Simulation Environment provides the designer with the required capability to perform the design tradeoffs. The first step to creating the environment is defining the hierarchy of the systems and subsystems. The disciplinarian models developed by the experts must then be integrated. The architecture baseline is then modeled and parameterized. Surrogate models for sensor error estimation are created from the model. The decision making algorithms are incorporated and integrated into the environment. At this point, the environment contains surrogate models of the sensor's error, disciplinarian models of the physical environment, and decision making algorithms. The final portion is the modeling of the scenarios that will be used to study the behavior of the system. In the case of the IEP it could include a vulnerability study that would disable portions of the system to simulate battle damage.

Optimize the System

The enormity of the solution space requires the designer to review an impossibly large number of combinations. For this reason, the process must be accelerated, as it will be discussed in more detail further ahead in the report, there are two options, increasing the computational power or create simpler models that describe the behavior of this more complex model accurately. The operation to be conducted on the simpler model is the optimization of the system's behavior by varying sensor types, placement,

network layout and control strategy. This provides a holistic solution that encompasses hardware selection and placement, communications, decision making and disruption of all of these.

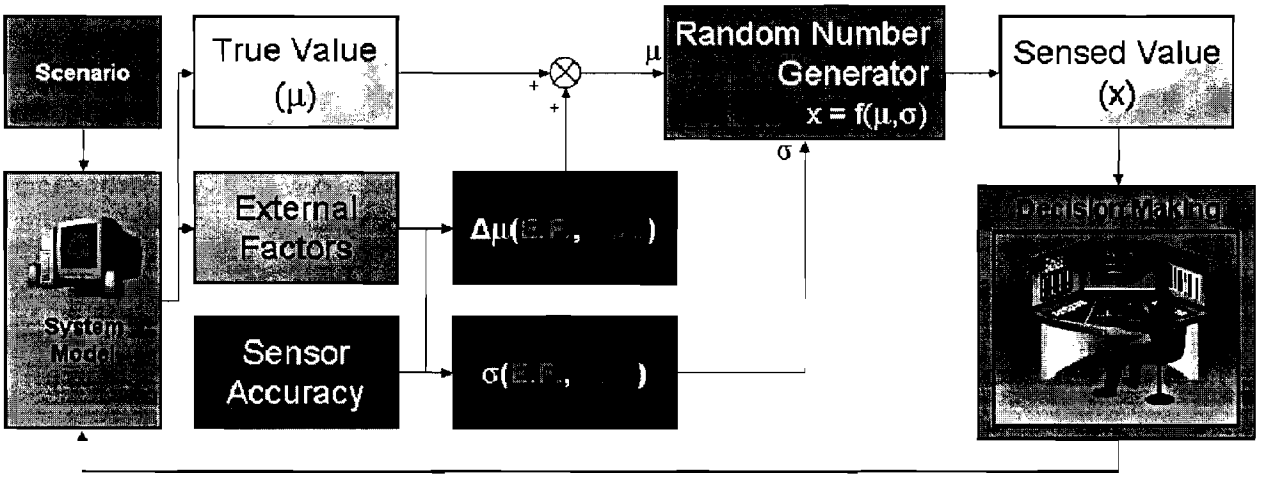


Figure 8: Stochastic sensor analysis process

The process presented in Figure 8 describes the steps that must be followed to model the effect of sensor accuracy on the behavior of an IRIS system. The first step is feeding the scenario to the model of the system; this could be the model of a ship, aircraft, network, etc. The values from this model are queried and stored as the true value and external factors that affect the sensor sensitivities. The sensor accuracy is obtained from the type of sensor used. The external factors and sensor accuracy is used to compute the deviation from the mean and standard deviation that the sensor reading could experience from the given conditions. This data is used to generate a random number which becomes the simulated value submitted by the sensor to the decision making algorithm. The decision from the decision making is used to actuate on the plant, modifying the behavior of the system model. This loop must be iterated continuously throughout the simulation for every sensor in the grid. The assumption that the sensor error is normally distributed is soft in the sense that the distribution could be easily modified (as long as it is possible to characterize it fairly easily), but empirical data shows that the error follows a normal distribution in the majority of the cases.

5.1.2 Design for Mission Effectiveness

Survivability is the most critical factor in determining the overall mission effectiveness. The formal definition of survivability is given as follows [Mavris and DeLaurentis, 1995].

$$\text{Survivability} = 1 - (p_D \cdot p_{H/D} \cdot p_{K/H}) \tag{1}$$

where the probability p_D is the probability of the system being detected, $p_{H/D}$ is the probability of getting hit if being detected and $p_{K/H}$ is the probability of being killed if had gotten hit.

Reconfigurability is another mission effectiveness factor that demonstrates how can the system "reset" itself after being affected by a set of events that occurred. Both survivability and reconfigurability contribute to the overall system mission effectiveness.

The inherent reliability of the system is also a major factor that can affect system mission effectiveness. Fault tree analysis is one of the most popular methods for system reliability assessment, where the system is actually broken down to its components at a lower level. Additionally, every component can contribute in its own way to the top level system reliability and by combining these individual effects according to the system breakdown, one can obtain a top level image on the overall system reliability behavior. The formerly described tool is primarily used for static systems; whereas the Markov Chain method or the Petri Nets are more suitable methods for dynamic systems, since they provide the capability to monitor the system status over time [Volovoi et al., 2004].

The methodology for improving the system mission effectiveness should come after the system is optimized in terms of technical feasibility and economic viability. Just by looking at the system from a reliability standpoint, reliability assessment methods typically require complete designs in order to be applicable. The methodology consists of three basic modules:

- Failure event consideration along with damage scenario modeling
- Modeling and simulation environment
- Design space exploration associated to the mission effectiveness

Damage scenario modeling is a necessary process that will allow the current system design to be exposed to a set of events that affect its mission performance. There need to be a few assumptions for constructing an automated damage scenario generation module. The goal for such a module would be to create an environment where inputs such as, type of missile or mine used for the attack, location of the attack or magnitude of the explosion, would be enough to generate a damage scenario that will include the physical system components that are affected by the attack, how these are affected in terms of their functionality and eventually at which time instants all these actions happen. The above ideas have all been combined into a well defined process, as shown in Figure 9.

It is necessary to obtain some insight on what the consequences of a hit on the system can be. In other words more knowledge is required as to what damage can occur to the system after a given type of hit and how damage will propagate through space and time further within the system.

Damage propagation from a hit location to a component and furthermore to other components depends on the component connectivity and change in environmental conditions. In order to increase the knowledge about how damage after a hit is being propagated, certain information concerning the physical topology of the system is required. An algorithm that reflects a process of identifying components that are

affected by the attack should be developed based on a specific logic. Moreover, damage propagation is a highly uncertain process. Certain types of information can be employed for predicting the magnitude and the location of the hit, as well as the type of damage that will occur and how these will expand to neighboring components. Finally, damage propagation is a dynamic process. Since it is a process that propagates through space from the hit location to neighboring components, it is certain that a finite time period is required for such a process to occur. Properties of components affected by the hit will change over time and every new state of the system should require updates from the control system on how to reconfigure the system and ensure maximum survivability.

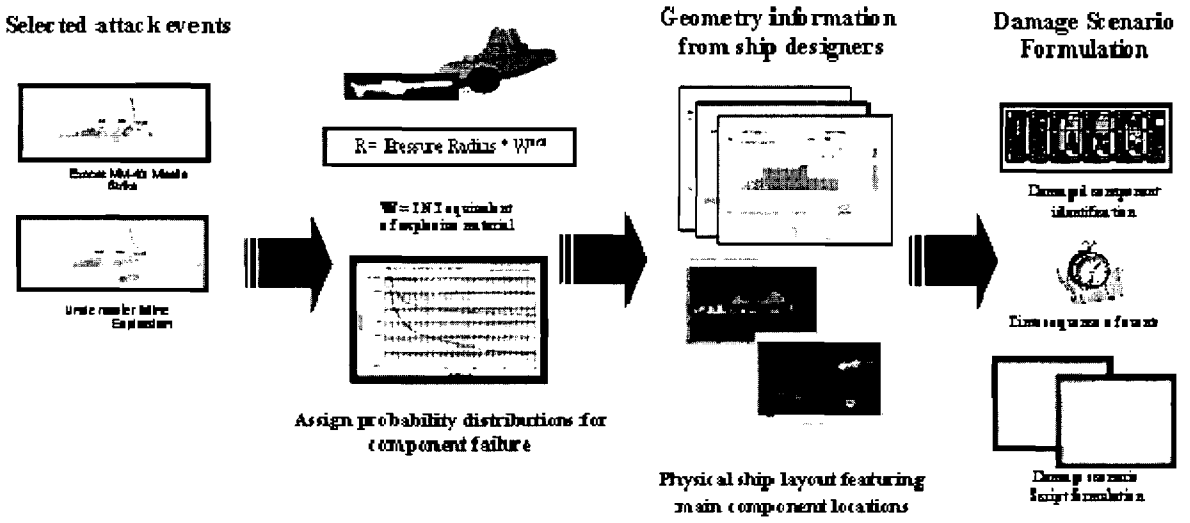


Figure 9: Damage Scenario Modeling Environment

To summarize, damage propagation among interdependent systems mainly depends on:

- Subsystem or component damage with/without subsequent failure. The failure of a component will then determine what the operational status of this component is and that the control system should be aware of. The process that leads to a failure from damage in a component is mainly determined by the characteristics of the component itself
- Change in system environmental conditions due to damage events in neighboring components (e.g. fire, smoke, flooding). A separate model can be developed for the study of fire propagation and how that affects the environmental conditions inside the system
- Further damaged and/or failing components due to the change in environmental conditions

A simple mathematical/geometrical model for identification of damaged and/or failed components has been created by ASDL as a starting point for a more sophisticated model that will have all the attributes required for more accurate prediction of how damage propagates in physical space and time.

A specific topology for the components is assumed, however without mapping their interdependency at this point. In other words a reference system has been defined and every component location on this system is known. It is also assumed that after a hit at a specific point on this topology, components that lie within a finite volume that has the shape of an ellipsoid are the only ones that can be affected by damage and furthermore be subjected to failure. This model will have to make a decision about how damage propagates within that volume, which components are damaged and which will eventually fail.

Therefore, the model is performing the following actions:

- Define a notional ellipsoid, centered at the point where a hit on the physical layout is assumed. The dimensions of this ellipsoid can be determined by processing historical data that refer to similar type of hit. In a dynamic version of this model such estimate can be used as a starting condition
- Locate ship system components inside the ellipsoid. If a component lies partially within the ellipsoid, it is assumed to be fully inside the damage volume
- Make a decision for the component status regarding their damage and/or failure
- The status of all ship components after a damage incident will then be used as inputs for the other models to run the M&S environment

The process that the current version of the damage propagation model is performing is as visualized in the following figure.

However, for a more reliable prediction of how damage is propagating, an accurate ship system topology is required to:

- include system component interdependency
- calculate changes in environmental conditions
- Capture damage propagation over time and reduce the uncertainty related to how damage can propagate from a component to another neighboring component.

It is quite understood that exact information for this purpose is or might be proprietary, therefore fictitious physical layouts are currently used.

As mentioned previously, damage propagation is a dynamic and highly uncertain process, as it involves a time evolution of events that are not uniquely and deterministically defined. Probability theory should be employed to capture the inherent uncertainty of the process, especially the aspect of the process that is associated to decide upon the status of every component. Probability distributions that are defined that are defined for that purpose should also be allowed to change in time as environmental conditions are dynamically changing as well.

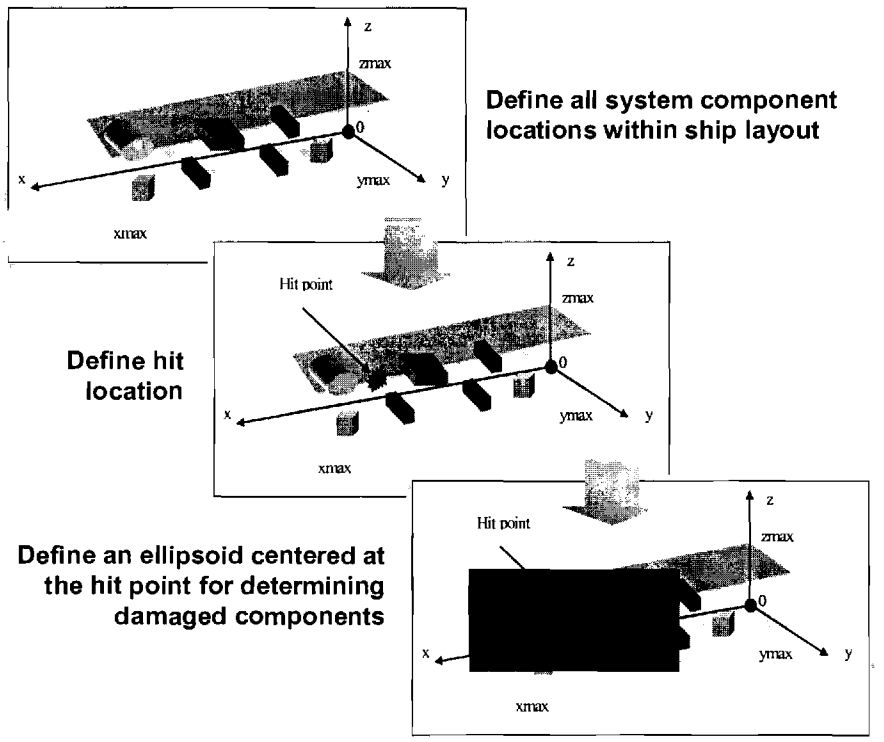


Figure 10: A simplified damage propagation model for integration in the IRIS M&S environment

For the current version of the model, time-fixed probability distributions are defined where the probability of failure of a component at a given distance from the hit location is given as an exponential function of this distance. This is a very simplistic approach, but for the future development of the model, the following enhancements will be considered:

- Suitable Probability Density Functions (PDFs) must be selected in order to create a 3D Joint probability distribution inside the ellipsoid. Damage propagation is a 3D phenomenon; therefore a 3D probability distribution changing with time and space is required.
- Factors such as environmental conditions (pressure, temperature, humidity) and the presence of structural components (bulkheads, compartments) can be represented by the variation of the 3D probability distribution and reduce uncertainty in decision making for a component's operating status.
- Additional information concerning the type of the hit by including historical data (type of attack equipment, strength of hit) for more accurate damaged area prediction (size and/or shape).
- Definition of the point where the ship is assumed to be hit. Should this come out of a random point assignment or from a more sophisticated subroutine that will take into

account locations on a naval ship that are strategically more probable to be hit by an enemy?

- Definition of inner hit points if the external hit location on the ship is known, according to the penetration capability of the object (missile, torpedo etc) used for the attack. Accurate ship layout information is again required to associate externally hit points to the corresponding internal.

A map then is needed to be able to identify what subsystem is located at every point of the ship layout. After the hit point is specified along with the associated damage radius, this map would reveal the components that should become affected by the hit due to their position. This information will come of from naval ship designers.

The damage scenario will then be constituted by the upcoming events, just from a simple initial hit event. Since these events along with their time points are defined, then this process can automatically create the script that the modeling and simulation environment would require for evaluating the mission effectiveness of the system.

The modeling and simulation environment for developing the formulation of the methodology of design for mission effectiveness is a complete power generation and distribution system based on a naval ship, known as Zonal Electrical Distribution Analysis (ZELDA) and was developed by Anteon Corporation in the past, for conducting fuel consumption studies. Although it was not developed for the purpose it is being used now, many changes have been applied to create a suitable model for initiating the development of the method. It will be seen that it is not the most proper model for this purpose but it is good enough for setting up the basic elements of the methodology.

Design space exploration based on mission effectiveness will be the analog to feasible or viable design space exploration that are related to constraints of technical or economic nature respectively. In this methodology, the space exploration will be associated to constraints that are determined by the desired or required tolerance to failure events and possible damage incidents during typical mission scenarios. The appropriate design variables should be identified to be completely able to define this design space. In the following there will be some preliminary suggestions on possible design variables and response metrics for quantifying and visualizing the impact of the constraints. Constraints that are expressed in terms of load power satisfaction or mission effectiveness will be dependant on the damage scenario cases that will be selected. Design of Experiments (DOE) can be used for obtaining the most useful and wisely selected scenario cases.

Four different types of design variables were identified and are illustrated by the figure below

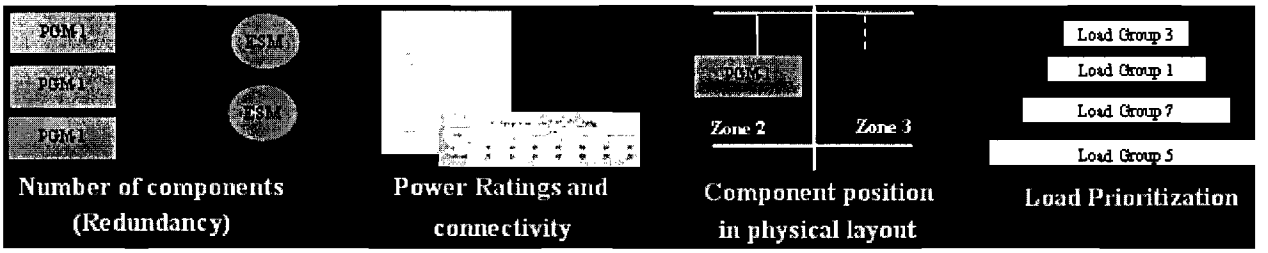


Figure 11: Proposed Types of Design Variables

The main concern would be to identify an optimized baseline that allows for the system to remain less affected by the occurrence of combinations of failure incidents and maintain the "non-failure design space" as large as possible.

Redundancy is associated with the ability of the system to reconfigure after being affected by certain failure events, whereas *power ratings and connectivity* will determine the levels of load satisfaction and therefore contribute in mission effectiveness. *Component position in physical layout* will allow for spatial configuration studies for a topology that assists in preventing failures after a damage incident. *Load prioritization* is also associated to reconfigurability and will be useful for operational studies and the implementation of a power advisor type of module, for bringing automation in decision making into the system.

Mission Effectiveness is defined as a percentage and can be calculated according to the following formula that ASDL has introduced

$$Eff_{Subsystem} = \frac{\sum_{Subsystem\ loads} w_i L_{Ai} / L_{Di}}{\sum_{Subsystem\ loads} w_i} \tag{2}$$

where, LA_i is the actual power consumption of load I and LD_i is the demanded power by load i. Maximum effectiveness (100%) of a particular subsystem is achieved if the supplied power to this subsystem is equal to the power it demands. The coefficients w_i are weights for modeling the importance and priority of every load and the weighted sum of all the subsystem loads is used for normalizing the effectiveness metric. While this is mission effectiveness metric that is solely based on power availability and thus is partially representative of the systems capability to perform the mission, it has been used successfully in the initial formulation for this methodology as a simple proof of concept.

It should also be noted that this method when integrated with the TIES, should form an overall process that can be iterative, in order to reach a convergence point that will be able to provide the best design choice, covering all technical, economic and failure robustness design constraints.

5.2 Model Integration

The main objective of the IRIS framework is to provide a solution for the IEP that can ensure increased survivability characteristics and autonomous decision making capability, along with significant manpower reduction for the ship's operations. The process will involve the design of large dynamic networks, consisting of highly complex systems that will probably demonstrate high levels of interdependency among each other. Typical examples are power system components, or ship service loads that may be coupled with parts of the cooling system or the sensor grid.

The development of an integrated modeling and simulation environment involves different disciplines, linked together either physically or theoretically, along with their associated time dependencies to reflect the dynamic nature of a ship's operations. Due to the system's complexity, and since every component can be considered a system itself, it is advisable to take a system-of-systems approach. Finally, for a study of a system of this size, it would be impossible to proceed without the aid of a virtual prototyping tool, not only from the aspect of available financial resources and time, but also due to the difficulty of building a hardware prototype without prior knowledge of the system behavior.

The dynamic nature of this concept is another strong reason supporting the development of an integrated model. Synchronization of individual models with multiple time scales is the biggest challenge here, and methods are under development to ensure the harmonious interaction of the lower level models. This will add the capability of running event driven scenarios that can affect the operation of all systems, with the expectation that the system controls will also be able to drive and reconfigure the system in the course of the simulation.

This virtual prototyping tool will consist of models and software that will be used during the IRIS implementation studies before building an actual hardware prototype. This tool, an Integrated Modeling and Simulation Environment, is a virtual computational representation of an actual system. It consists of a set of subsystem models, properly combined and linked to each other, for the purpose of creating an environment to allow for the user/designer to simulate the operation of naval ship systems under given mission scenarios. The modeling and simulation environment can be leveraged to do several types of studies:

- Control as an independent variable – More precisely, system and resource management techniques and algorithms can be applied to the virtual system as an additional dynamically interactive component. A set of experiments can be run that focus on the

parameters of the management solution, or compare different control/management architectures.

- Design space exploration – through the use of designs of experiments, surrogate models, and statistical tools, designs can be evaluated or, using requirements/constraint-driven methods, can be narrowed down based on their predicted performance (i.e. inverse design).
- Reliability analysis – A given system can be simulated under a variety of initial conditions, to gauge its response to changing environments and/or a spectrum of damage conditions.
- Human-in-the-loop simulation – An interactive prototype "advisor" console could be used to test methods of presenting automated and assistive decision-making to the operator.

For a systematic method of capturing the system component interdependencies, a method called Interrelationship Mapping is proposed [Dieter, 2000]. In order to build an interrelationship map, the functional and physical system decompositions are the first performed in order to determine interdependencies between subsystems, equipment, and components at several levels of detail. This two-step process is also known as the "conceptual decomposition" of a complex system and will also be useful in the mapping of outputs of each subsystem component model to the inputs of associated dependent subsystems.

In the following, a set of different types of time-domain simulation integration schemes is presented, including the simulation scheme that has been adopted by ASDL for the integration of the models that will provide the platform for the ship operations' simulation.

5.2.1 Direct Translation

This approach consists of obtaining the mathematical models and programming them in the same platform. The benefit of this approach is that it executes faster than the other alternatives, the solution is more stable and the time integration of the simulation exhibits is more stable. The detriment to this approach is that the models need to be translated, involving a significant capital investment in terms of programming. Furthermore, disciplinarian modelers have preferred platforms on which they model their systems and many of these software packages do not make their mathematical models apparent, an example of this is FlowMaster2, which was used by NSWC-CD Philadelphia to model the CW-RSAD. This approach produces a tightly integrated model, as evidenced by the work performed by the team

developing the Nonlinear Control System (NCS) composed of researchers at the United States Naval Academy, MIT, Purdue University, and Anteon, Co.

5.2.2 Compilation

Model is compiled in the host environment. Modeling environment to be included generates a dynamic-link library (DLL) file that allows the host environment to execute the model. DLLs are an implementation of the shared library concept that allows multiple programs to utilize the same modules of code. DLLs may contain code, data and resources in any combination. The included model does not contain a solver and the computation is performed by the host environment. This is a feasible solution when the number of states is not excessively high and the simulation does not use internal solvers to determine the behavior of the system. As the number of states increases, the time step simulation should be paused to allow for the swapping of data, but at the same time, retain the information required to determine the state on the next time step. This method is less robust than direct translation, but more than co-simulation. At the same time it is considerably easier to integrate than direct translation if the modeling environment allows for the creation of the DLLs.

5.2.3 Co-Simulation

This approach sets up one software package as a master and the others as slaves. The different simulators use their own independent solvers allowing for more flexibility to the modelers and permitting more direct integration of the models. The biggest detriment to this approach is the instability produced by having collaborative solvers.

5.2.4 Multi-Disciplinary Simulation

Multidisciplinary simulation is a time domain simulation scheme for integrating different computational models and thus creating a modeling and simulation environment for modeling a naval ship's operations. It is a development of the co-simulation scheme, but has also some influences of Multidisciplinary Optimization methods and specifically optimizer-based decomposition. The OBD structure of passing information among the models is maintained and a time synchronizer plays a role similar to the one of the optimizer at the top level.

In order to apply MDS, system decomposition is necessary for obtaining the system component hierarchy. The first part of the conceptual decomposition, involves the breakdown of the complex system in the physical domain, decomposing it into its lower level components. Affinity diagrams and tree diagrams [Dieter, 2000] can be used for visualizing the results of this process. Figure 12 shows a tree diagram representation of the physical decomposition of a notional system consisting of a computer

coupled to a thermostat-controlled coolant pump; this system serves as a test platform for initial modeling and integration studies.

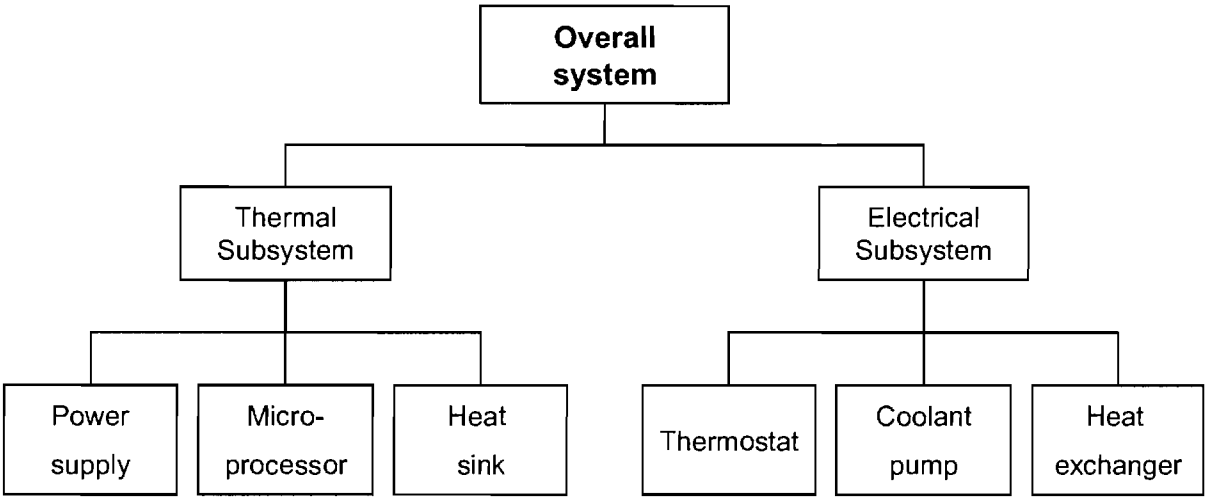


Figure 12: Tree diagram for demonstrating the physical decomposition of a thermal/electrical system

The second part is to identify what functions these components must perform to allow the total system to operate as expected. This is known as the functional decomposition, where every component is mapped to the set of actions that it is taking or, its physical behavior, during a period of system operation. Functional decomposition of a system can be documented through a table which includes a set of information for every subsystem/component into which the top level system is broken down. The physical inputs and outputs need to be identified (along with a corresponding metric if possible) as well as the function that is performed within the module. Other side effects can also take place while the function is performed and these are documented as secondary functions that have their own responses to the same inputs. A demonstration of the functional decomposition of an electrical/thermal system is shown in Table 1.

In a highly dynamic system, the physical decomposition should not be different than the process applied in a static system. However, the functional decomposition will require additional mapping of the event sequence and the time intervals between actions, given the fact that the systems concerned are not only highly complex but also time-dependent. Events and functions performed may have a specific sequence and may have time intervals between actions. A flowchart similar to the one shown in Figure 13 can be used for this purpose. Events or functions performed by a component are denoted with a circle and the arrows represent the activities that lead from one event to another.

Table 1: Functional System Decomposition.

System	Component	Input	Output	Function	Side Effects
Electrical	Power supply	Mechanical Power	Electrical Power	Deliver electrical power to load	Produce heat
	Microprocessor	Electric Power	Heat	Perform calculations	Produce Heat
	Heat sink	Heat/ Coolant flow	Heat	Extract heat from processor/ power supply	None
Thermal	Thermostat	Temperature	Control Signal	Control operation of coolant pump	None
	Pump	Electric Power	Coolant flow	Circulate coolant	None
	Heat exchanger	Heat	Heat	Exchange heat with environment	Increase temperature

Diagrams such as the previous one are not only useful for mapping event time sequences and activity durations, but also allow for activity optimization, where functions can be further simplified, redundancies and thus costs of performing functions can be reduced. In a modeling sense, additional modules not only mean higher complexity but also require extra simulation time to run a case.

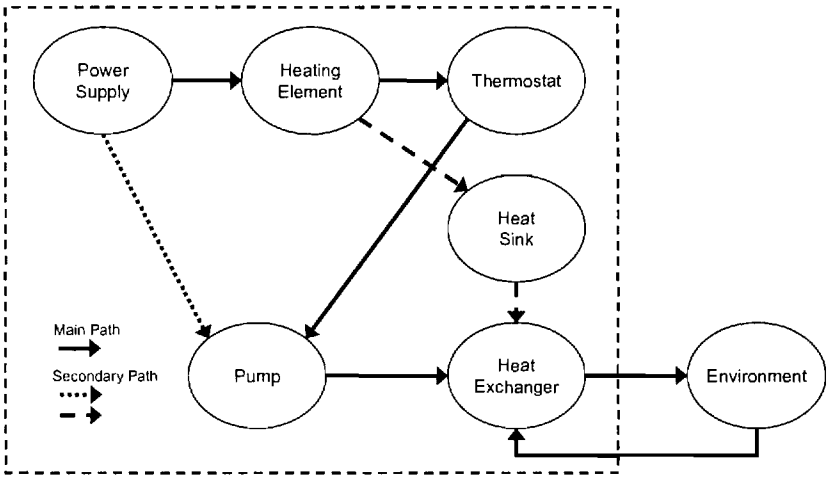


Figure 13: Network diagram for mapping of the event time sequencing

The overall system function can be viewed as the result of having every physical component performing its individual function according to the hierarchy and time sequence of events that is defined by the physical decomposition. This is the actual outcome of the Interrelationship Mapping process and the result for the thermal/electrical system is shown in Figure 14.

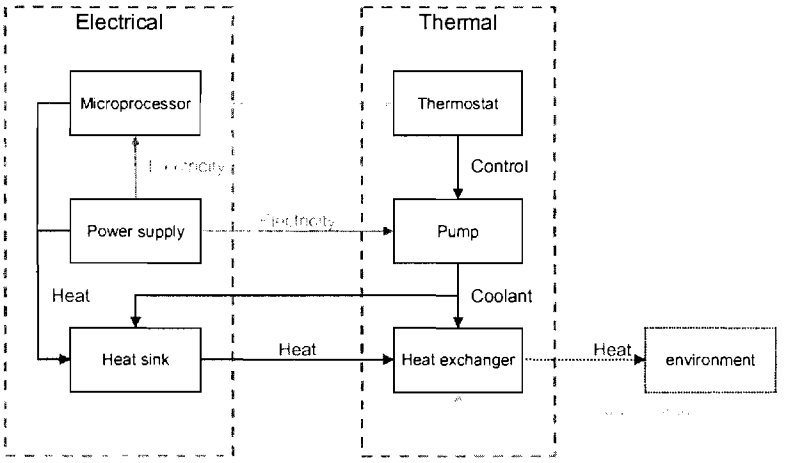


Figure 14: Interrelationship mapping for the overall system

It should be noted that, the way that system components are physically and functionally interrelated is not necessarily unique. On one hand it is desired to have a simpler and leaner configuration with fewer interdependencies and less complex connectivities, for improved computational performance and reduced operating costs. However it may also be desirable to have a design with more interdependencies and functions that need to be captured in order to develop a more accurate integrated model that is closer to reality. In this case though, more interdependencies will increase redundancy for the connectivities and for the event time sequence complexity.

The introduction of an Interrelationship Mapping strategy now prompts definition of a method and tools by which this can be put into practice. Multidisciplinary Engineering has introduced novel concepts for preliminary analysis of integrated system-of-system design. Multidisciplinary analysis (MDA) [Acton and Olds, 1998] [Browning, 2001] allows an integrated product team to make contributions from various fields (e.g. structures, hydrodynamics, sea-keeping, signatures, operations, etc.) and deliver a more complex picture of how the interdependent systems function. Figure 15(a) shows this organizational schema illustrated as a Design Structure Matrix, with notional processes A-D linked with a set of feed-forward and feed-back relationships. Typically, computation of results relies on numerical solution methods such fixed-point iteration [Chapra and Canale, 1998]. These results, either single responses or functions thereof, can be used as a set of evaluation metrics.

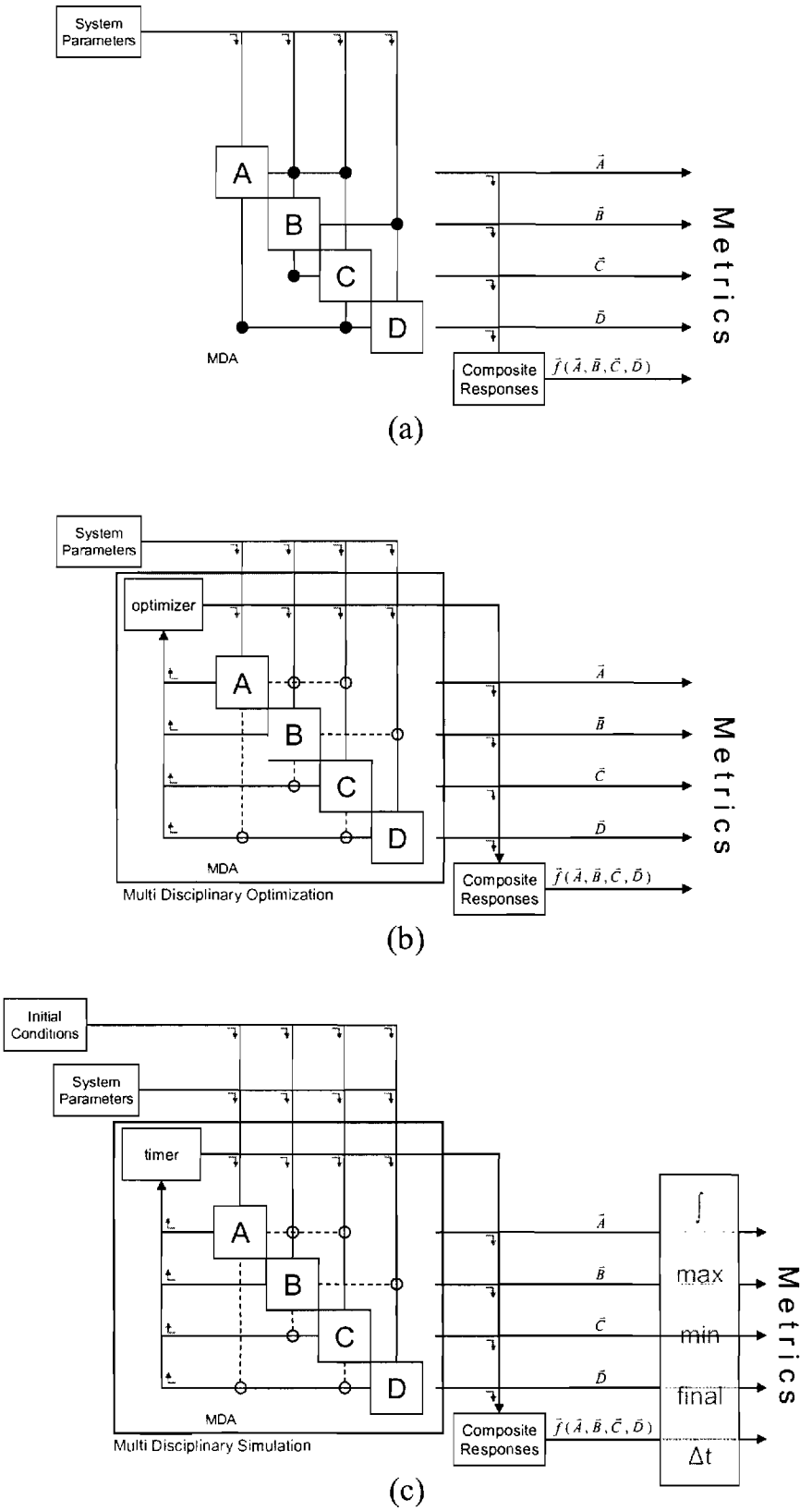


Figure 15: Derivation of MDS schema from multidisciplinary methods

Since analyses of complex dynamic systems such as naval surface warships will most likely include time-domain simulations, it is necessary to further extend the methods of multidisciplinary analysis. The concepts that underlie multidisciplinary optimization (MDO) [Kroo, 1997; Hulme and Bloebaum, 1998] [Cormier et al., 2000] especially decomposition-based methods [Ledsinger and Olds, 1998] can be applied to creating an organizational structure for joining together and running dynamic models of subsystems. Figure 15(b) and (c) demonstrate the adaptation of existing multidisciplinary methods to the application of integrating heterogeneous time-domain analyses. Instead of an optimization algorithm as shown in Figure 15(b), an external clock is used to synchronize the execution of the models, and a vector of initial conditions provides information for calculation of the first time step as in Figure 15(c). Metrics may also be distilled by calculating properties of simulation results, such as integrals, maximum or minimum values, time intervals, etc.

Using an optimizer-like strategy has several benefits:

- Control is maintained over the parameters that influence how the integrated simulation is scheduled and executed.
- As in optimization-based decomposition, the MDA derived from interrelationship mapping can be parallelized.
- Different versions individual analyses can be exchanged without having to reorganize the entire simulation framework.

Iteration through the desired time interval yields a time history of model outputs and/or composite responses obtained from functions of the individual output vectors.

Although current proof-of-concept models are simple enough that complex data handling is not required, the amount of data that will be involved in running simulations of complex, interdependent systems will be considerable. Especially when time-domain models are run, a system for organizing, transmitting, and storing data will be extremely useful. To this end, one option would be to set up a customizable, dynamically accessible database or file system that all relevant applications could communicate with and extract data from. This could be as simple as text or XML files, or a more complex Excel- or SQL-based database.

Regardless of the method employed for managing data, a backplane for linking all the models and their associated inputs and outputs is necessary. Since many analysis codes are full-featured standalone design tools, it is necessary to provide conduits for execution and data acquisition. Typically this is done using shell scripting that accesses a given analysis application via COM or OLE objects, or by a developer-provider application programming interface (API). The data backplane, models, and other tools can also be implemented in a commercially available process integration platform.

The last hurdle of this area involves coupling time-domain simulations and running them in parallel; for example, how can a model of an electrical system be coupled to a simulation of a fluid thermal management system? The main issues to be addressed include synchronized execution, and dynamic updating of parameter values. Work in this area is ongoing; to date the models used in this study have identical timescales.

A Simulink™ model of an electrical/thermal system was created that would exhibit time-dependent, nonlinear behavior to mimic what will be observed in an integrated naval system model. A process integration tool called ModelCenter™ published by Phoenix Integration [Ng et al., 2003] was selected to provide the backplane for integrating individual models, storing and exchanging data, and scheduling and driving execution of analysis codes.

To enable ModelCenter to handle dynamic models, a custom component called 'Simulator' able to drive the simulation using ModelCenter's data and scheduling infrastructure was written. It is a Java class that is called by ModelCenter, and uses the Phoenix Integration Java API to interact with the simulation components specified by the user. The user interface is a drag-and-drop interface that specifies links between input and output parameters for each code.

Data validation and computational speed were the two most pressing performance aspects that had to be confirmed before using this tool for future studies of actual research codes. Validation of data was done by running the test simulation (the thermo-electric Simulink code) in a variety of configurations; the data matched very closely between all the methods implemented. As shown in Figure 16, the differences between the original total system simulation and the ModelCenter-based multidisciplinary simulation are negligible.

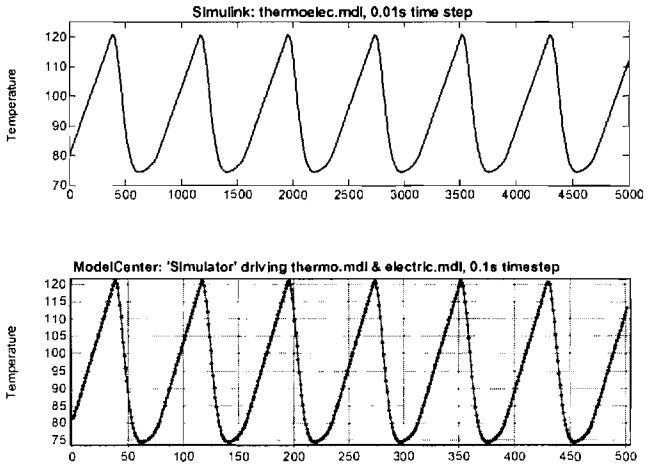


Figure 16: Results from the MDS method show data validated against a total system simulation

Computational efficiency turned out to be within acceptable limits for the simple model studied so far. Using the complete thermoelectric model, execution time ranged from real-time ($\Delta t = 0.01s$) to 10x

real-time (Δt = .1s). It was observed that running the simulation using the linked-code method driven by the Simulator component, required 30-45% more execution time.

However, it is still unknown how well the linked-code ModelCenter simulation method will scale to larger models or to having more linked simulation codes. The experiments performed thus far have used only two models with a total of eight state variables; a fluid system model alone may have hundreds of state variables. It has been shown though, that it will be possible to utilize a process-integration tool (e.g. ModelCenter) to link and automate the execution of simulations. Furthermore, if load distribution strategies (such as utilizing Centerlink, Phoenix Integration's job scheduling tool) can be used, or if analyses can be dedicated to certain workstations connected to the ModelCenter network, faster-than-real-time simulation of complex systems may be possible.

5.3 Modeling Physical Systems

There are three main systems that constitute an IEP and are included in the IRIS framework. An electrical system is necessary to generate and distribute power for mobility and ship operation purposes. A fluid system is vital for the protection of the electrical system from overheating and for supporting ship service and safety operations. A control system is required for facilitating mission related ship operations and ensuring mission effectiveness and enhanced survivability and reconfigurability.

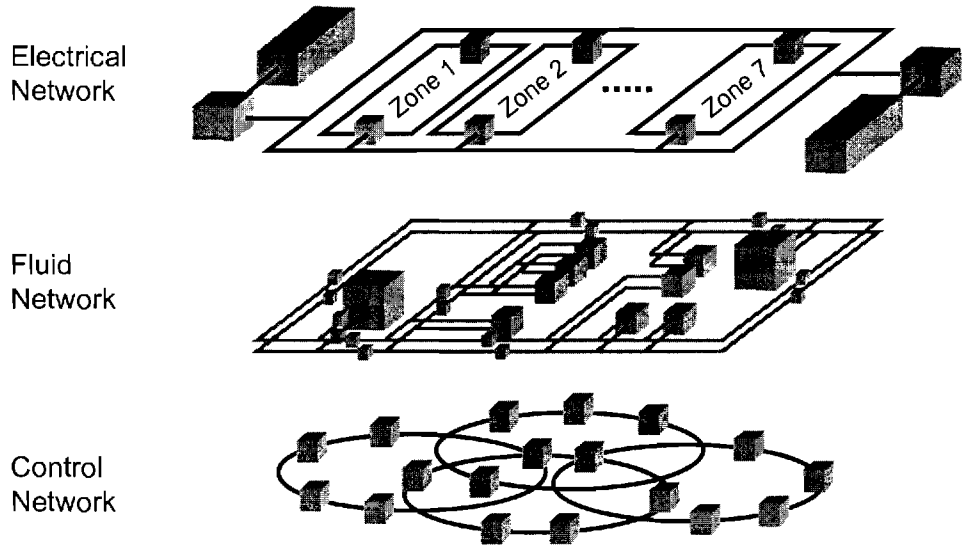


Figure 17: Networks of the IEP.

5.3.1 Electrical Modeling

Two approaches were pursued for the modeling of the electrical system; one aimed at performing a fast power balance and distributing the power to the service loads that require those resources. The other aims

at modeling the electrical system in a more physics-based sense for better accuracy in determining the top level system performance.

5.3.1.1 Low-Fidelity Electrical Model

Any physical system can be modeled using various levels of fidelity. Lower fidelity models run faster, are easier to integrate in multi-model simulation environments and are simpler to update, modify or append. They lack the accuracy because not all the physical equations are represented. In contrast, a high fidelity model represents the actual physical system. Its level of fidelity depends on the accuracy of theory that describes the system and the number or detail of implementation of the theory equations. Usually, an expert is needed to build high fidelity models. Also, high fidelity models are complicated, more difficult and time consuming to build and are not integrated easily in a modeling environment. The IEP requires both kinds of models: a low fidelity model for initial design and setting up the environment and a higher fidelity model as the design matures.

Two notional simulation models for the electrical power system are available for research. The first model, referred to here as the NCS model, was built using the Advanced Continuous Simulation Language (ACSL), which is a well tested and time-proven simulation environment, used in the solution of large systems of nonlinear ordinary differential equations. The drawback of this model is that it was built with ACSL v11, which does not have a graphical user interface. The model is not transferable to the newer version ACSL Xtreme, which is a well equipped with a graphical modeler tool. This model lacks the flexibility and portability needed for the development and integration of new components easily. In addition, ACSL requires extensive experience with programming and simulation environments. This prohibits the use of the model in the design process, since every minor change is usually accompanied by a tedious programming task. Also, the level of detail of components is very high, hence pulling down the model execution time. On the other hand, the model does not encapsulate many component types. The NCS model does not have the broad coverage of ship components, and has unneeded details of particular components. After two years of struggling with the model, it was concluded that ASDL cannot use it in its present form. Current efforts are being undertaken in translating this model to MATLAB™, as will be discussed in the following section.

The second electrical power system model available is ZELDA, developed by Anteon. ZELDA was originally developed as a fuel consumption tracking model for an electric ship. It is based on a realistic ship power distribution network, with actual components found on a ship. It encompasses about 300 loads arranged in seven zones. The zones correspond to a spatial layout. ZELDA already has the power assignment analysis done. It uses the simple concept of assigning power to a certain load, without

worrying about the physical implementation of this assignment. It is assumed that the components are well designed such that they would take care of their own physics.

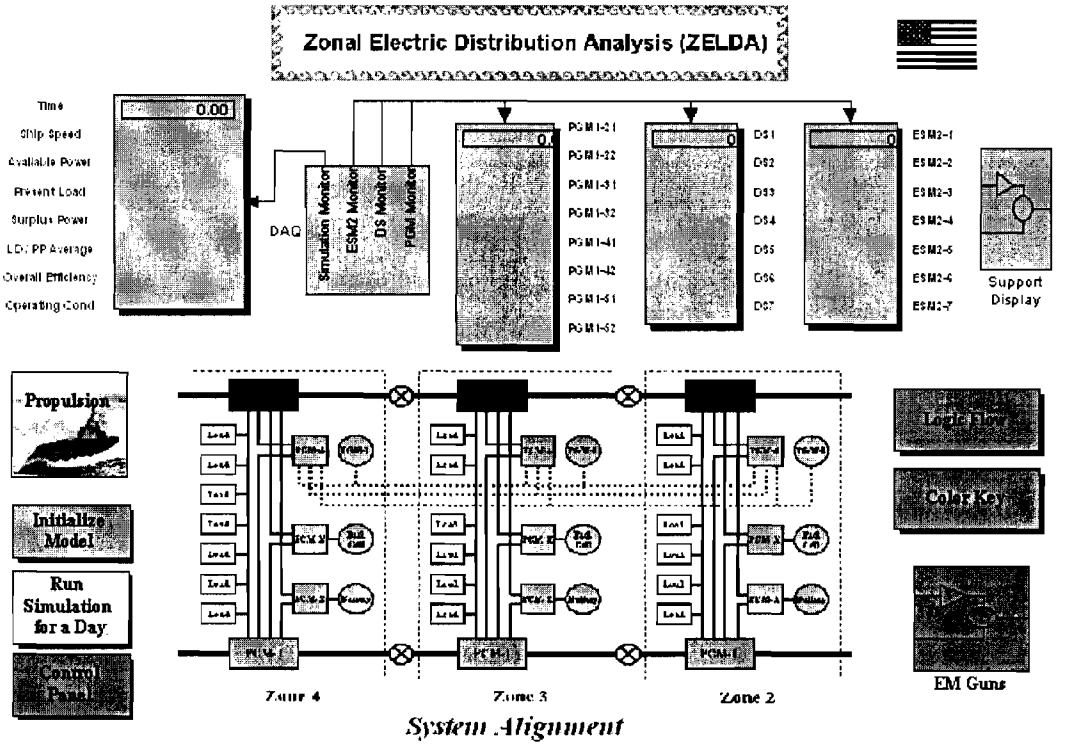


Figure 18: The ZELDA Power Model

Even though the ZELDA model seemed more suitable to the current research requirements, it has some shortcomings. It is built as one unit. Application of any of the features of the ideal model requires the modification of most of ZELDA. Moreover, it is an analysis tool rather than a design tool. Inputs are static, meaning that they cannot change during the run-time; they have to be preset. Due to the extensive use of some types of Simulink blocks, its compilation is not possible. This jacks up the overall simulation time making it difficult to run design of experiments cases or integrate in a multi-model simulation environment. Resource allocation algorithms and control over how power is distributed among the components is restricted to the use of priorities or price index method, and is hardwired. No new automation or resource allocation schemes can be tested unless a fundamental modification of the model is completed, almost as time consuming as rebuilding the model. Finally, although loads are placed in their corresponding physical zones, yet a complete spatial layer is not present. A network layer is not present either so ZELDA does not account for component interdependency. Hence results from different resource allocation algorithms and control algorithms are not accurate.

It was obvious that a model similar to ZELDA, yet avoiding its shortcomings, is needed for a low fidelity model. The original idea was to cut some corners from ZELDA to end up with the needed model.

Corners turned out to be chunks and the functionality of the model was affected. Instead, a different approach was taken. The logic behind ZELDA was studied thoroughly, and the part of this logic that suits the needs was used in building the new model.

The model should have an acceptable number of component types to study their interactions. A network layer is built in the model allowing for component interdependency. This network layer can be easily updated or modified to represent new designs by using a graphical user interface. In addition, an interface for resource allocation and control algorithms was standardized. A priority based resource assignment system is used (discussed later in the implementation section). Any resource allocation algorithm controls the model by assigning a set of priorities to components, hence shedding loads, shutting down load sub-networks or re-routing power. Separating the resource allocation algorithm form the actual model itself gave way to a multitude of algorithms that can be tested, ranging from agent based to neural nets. Distinction and separation between the control network layer, the power distribution layer and the spatial layer is essential.

Although the initial model is a low fidelity model, various levels of fidelity are easily added to the architecture. Eventually, this leads to a mature high fidelity electric/power model. Increasing the fidelity also includes designing place holders and interfaces for other sub models, such as a damage model, a flow model and a spatial such that these sub models can be added at a later stage.

Object oriented / object based model design methods were used due to their effectiveness as will be seen later. Physical systems can be viewed as the collection of components or entities whose interaction with each other determines the behavior of the system. These entities are referred to as *objects*. An object consists of two main parts: properties; data that describe this object, and methods; things that this object can do. Because the object's properties are only accessible through the object and its methods, objects are self contained. This is referred to as encapsulation. This makes it significantly easier to work on parallel computing platforms, decreasing the simulation time. Object oriented /object-based simulations are very common and appealing in modeling and simulation of physical systems, hence they were used in the present research. This is contrasted to process simulation, in which complete processes are modeled rather than components. According to a predefined logic, execution of the simulation moves from one subprogram to another till the simulation is terminated.

Simulink / MATLAB™ was the choice tool of implementation. It allows for both a graphical interface and an object based design. That is in addition to portability of models to different platforms, the ease of use, and lower modifications and update times.

5.3.1.2 High-Fidelity Electric Model

The electrical system of the IEP architecture is expected to be represented by an electrical power generation and distribution system. Not only will it include the modules that describe the generation of AC power (prime movers, synchronous machine, etc...), but also a zonal architecture will be constructed for the propagation through the buses and conversion of this AC power in the zones, so that the power reaches the recipient loads (service loads and propulsion units). The desired electrical model therefore should be dynamic and capable of predicting the power that goes to every load, as well as the power losses at every time instant. Power losses that are converted to heat, is necessary information to be propagated to the fluid and control systems for preventing component overheat and ensuring proper cooling.

Low level system calculations, such as voltage or current histories for the components are not explicitly required. However, such time histories are important for obtaining accurate estimations of power losses to monitor the amounts of heat produced by power consuming electrical components. For the IEP subsystem model integration purposes, all that is required by the electrical system analysis model is to provide with the time histories of the power flow from the generator to the service loads. At this point the prerequisite capability and fidelity of the analysis tool for the IEP electrical system has been determined. Next, the challenge was to search for this tool and selecting the most appropriate one from the list of the available tools.

The fact that two different models are available to ASDL to be used as an electrical system model for integration purposes does not mean that these models compete with each other. A pure physics-based high fidelity model and an electrical power management model can be modified properly to be in a form that they can eventually be integrated into a unique module for an electrical power generation and distribution model. The NCS model can be used as the basic "map" for understanding the physics of its operation. Such knowledge can useful for translating and rebuilding a simplified version of the same model in an environment that will allow the user to manipulate the model and is more compatible with the "wrapping" integration software.

Purdue University in cooperation with the US Naval Academy has developed a sophisticated physics-based model of an electric power generation system for naval ships [Sudhoff et al., 2004]. It consists of a 6 layer layout:

- Spatial Layer
- Automation Layer
- Communications Layer
- DC Power generation Layer

- AC Power generation Layer
- Thermal Layer

where each layer is assumed almost independent of the others. Calculations are being performed in every layer, creating information that is passed to the other layers for local calculations. A library of functions is used for the purpose of activating a layer, so that it can do all the required calculations over time. Since the NCS model is a dynamic model, it has to perform time domain integrations of rates that change based on dynamically changing electrical system properties.

Finally, the NCS model was implemented in ACSL® v11.8 (Advanced Continuous Simulation Language) as a monolithic tool (modules integrated within the tool, without the use of any "wrapper" software application) for achieving better performance and computational speed with time consuming time domain integrations, as well as for improving computational efficiency.

The NCS electrical model is a well performed sophisticated physics-based model, yet not in the suitable form that is required for integration in the IRIS M&S environment. ACSL is an excellent choice for the implementation of such a model; however, there is some difficulty in altering the source code of the model to adjust it to specific IRIS M&S framework requirements. This difficulty becomes even more important considering the fact the "wrapper" tool might have its own requirements for being able to guarantee the smooth integration between different tools in an IRIS M&S framework. Another obstacle in the process of utilizing the NCS model is the fact that the ACSL software version (ACSL 11.8), at which the model was implemented, is and will be not supported any further by the distributing company.

The NCS electrical model does not include an Application Programming Interface (API). Therefore, it would be impossible to update input values in real time for subsequent simulation cycles. On top of that, the architecture of the electrical model itself is such that, even with the existence of an API it would still not be certain if the user could interfere and change input variable (status vectors) or system parameters. Such capability would be important for the case of performing design trade studies.

The NCS electrical model includes a thermal layer that is also modeling the process of chilled water circulation to transfer the heat that is being generated on the surrounding of an electrical component or load. A more sophisticated, complete chilled water system model (available to ASDL by NRL Philadelphia) has already been integrated in the IRIS M&S environment. Thus there is overlap between the thermal layer of the NCS electrical model and the CW-RSAD model (implemented in FlowMaster2™).

Eventually, it was decided that an effective strategy for using the NCS model would be to understand the physics behind it and construct a simplified version of the original model in a software framework that will be compatible with the integration environment. Visual Basic, Java or MATLAB™ were all good

candidates for the purpose, but implementing this model as a combination of MATLAB™ scripts seemed as the most convenient choice. More about this simplified model implementation will be discussed in a subsequent section.

5.3.2 Fluid Modeling

A software version of the Chilled Water Reduced Scale Advanced Demonstrator (CW-RSAD), which is a physical model of the DDG-51 chilled water system, was developed in Flowmaster2, a commercial fluid network analysis tool. It consists of sixteen service load networks and two pump and chiller networks, and the system is equipped with 72 valves to reconfigure the whole CW-RSAD network. Each service load network has a heat exchanger that cools one of the various systems in the ship, such as radars and threat receivers, navigation equipments, weapon systems, and crew quarters. In the integrated modeling and simulation, it is assumed that the CW-RSAD cools electric devices in the electric power system. This is not entirely accurate, since in reality there is an intermediary system that cools the electric power system, however that model was not provided. Also, there is insufficient information describing which facilities and devices are mapped to which of the sixteen service loads in the CW-RSAD model and how much heat these generate. The goal is still to show that the integrated simulation environment can manage the interactions among subsystems using the subsystem models that are available.

There are other fluid subsystems that comprise the ship system but were not thus far provided. As mentioned above, in reality the electric power system is cooled by the fresh water cooling system. There are also lubrication oil systems that provide synthetic or mineral lube oil to the power generation and propulsion units. Fuel service systems provide fuel transfer to generation units and are located in three main generator rooms in the ship. Finally, there is a sea water system to which the fresh water and lube oil dump the heat transferred from the various heat generating units. The sea water system also works as the auxiliary cooling system throughout the ship. In the current integrated model, the chilled water system has been successfully implemented, so that the procedures and methods developed for the integration of the current chilled water system can be used to integrate other fluid system models that will be provided in the future.

As previously mentioned, the initial fluid network model that represents the ship's chilled water system was provided and the main task was to modify this initial model to have the capabilities desired and to implement it in the integrated simulation environment.

5.3.3 Network Modeling

The onboard communication network is the enabling system to achieve the goals of the IRIS project. To be able to automate the sense, assess, react sequence, a robust, reliable and reconfigurable communication

network is required. Thus far in this document, the overall vision and motivation of the IRIS project has been laid out and the key concepts in communication network technology has been introduced. This section will describe in more detail how the networking technology will be critical and an enabling function in the IRIS project.

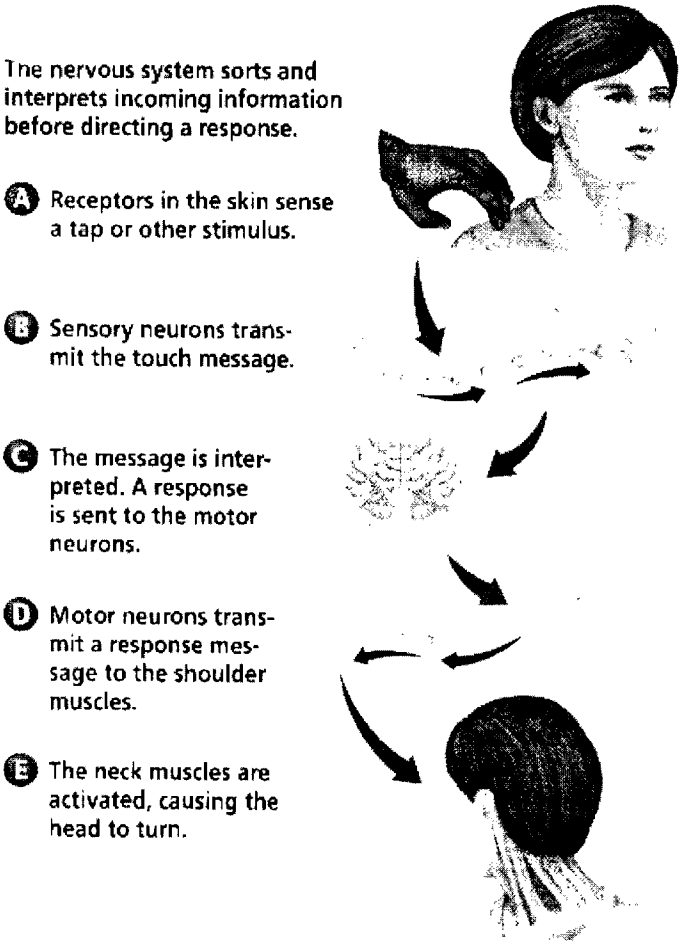


Figure 19: Communication Network Analogy to the Human Nervous System

The communication network is the backbone of the IRIS vision. It is the technology that will enable the desired capability of automating processes. In the scope of the IRIS project, the communications network is analogous to the human nervous system or the backbone. "The Nervous System is the body's information gatherer, storage center and control system. Its overall function is to collect information about the external conditions in relation to the body's internal state, to analyze this information, and to initiate appropriate responses to satisfy certain needs (Maintain Homeostasis)." [Johnson, 2004]. An example of this scenario is depicted in Figure 19. The human body is equipped with receptors or sensors around the skin. When a person is tapped, the receptors send a message through the nerves on the system to a centralized point, in this case, the brain. The brain takes the message, interprets it, assesses what has

happened and determines the correct response to the action that which has been receipted. The response is to turn the head in the origin of the tap. The message is sent to the appropriate muscles to physically move the head in that direction.

The relation of this analogy to a ship is depicted in Figure 20. When a ship is hit, the sensors along the structure and in the interior will be able to locate the point of destruction and assess the level of destruction of the ship. This information is sent to a computer where the information is assessed and the decision to how the damage can best be contained and corrected is made. This information is executed appropriately using the automated systems onboard the ship and the ship is contained and can function.

An understanding of evolution from current technologies to the desired functionality of the present and future technologies as well as their application to the IRIS project must be established. The Smartship program was a first attempt to automating the sensory capabilities onboard a ship. This project focused on applying technologies from industry and research to enable the optimal-manning operations in order to reduce risk and cost of acquisition of fleets. This project acts as an appropriate predecessor to the IRIS and IEP ship concepts because it began the background research and preliminary designs for automated ship scenarios [Smartship, 2004].

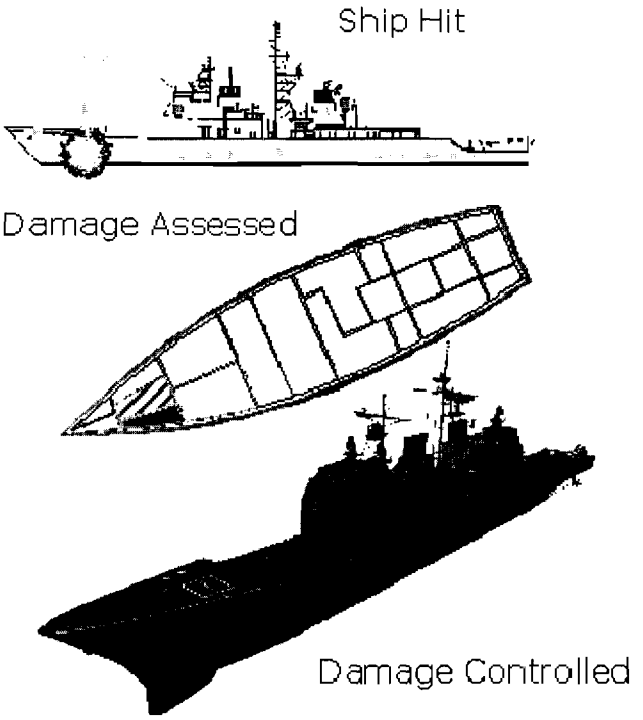


Figure 20: Human Analogy Relation to a Battle Ship

Fire extinguishing mechanisms is one of the established technologies that have been researched and implemented onboard the Smartship [Gillis et al., 2003]. The technology has been implemented to assess the health of the ship and to make decisions for automating the fire extinguishing mechanism. Some of the tradeoffs that were considered in this study were an in depth study of the potential threat of the damage by the fire currently being assessed and the damage that will be incurred by the method of extinguishing the fire. A variety of methods of extinguishing was examined and explored so that the automated system would have a reliable database to predict and decide the appropriate action for surviving the disaster. It is the concept of automating the process of risk based decision for damage control that was the novel achievement in this process and the communication backbone was established to enable this technology [Gillis et al., 2003]. The goal of the IRIS project is to perform a similar technology tradeoff study for other systems which can be automated.

The IRIS project will incorporate several technologies to achieve the robust automation decision making capabilities desired by the current thrusts of the navy. These technologies, as eluted to before, range from propulsion systems, structural engineered parts, automated controls and an immense set of sensors. The IRIS project is state of the art and will reduce the manpower required onboard a ship and thus the operational costs. The methodology will be part of the paradigm shift of design to revolutionize the design effort of the naval vessels. The consequences of this effort do not only affect the economics of the mission, but require advances in the communication and networking onboard the ship. The preliminary focus will be on the propulsion systems, navigation, health monitoring (personnel, equipment, and ship), weapons readiness, video system, fluid systems, damage control and fire detection.

5.3.3.1 IRIS Network Challenges and Baseline Settings

Along with the Smartship program, Anteon Corporation has been researching the problem of creating a sophisticated network to have the needed features for the IEP problem. Anteon's efforts lay hand-in-hand with the necessary framework for the IRIS project. They have researched the problem and began to specify some challenges of this problem as well as some characteristics. The suggested topology is a full mesh fiber optic LAN. This LAN would use Asynchronous Transfer Mode so that messages can be sent across the network and solutions can be created in real time [Dunnington et al., 2003]. The challenge Anteon has encountered includes reducing the number of dependencies on the network between the components on the system. A network with these types of dependencies will reduce the certainty that changes in the network will require no more time and effort than is reasonable and thus impeding the ideal reconfigurable solution. The network must be fault tolerant, highly reliable and survivable in reconfiguration. Minimizing the number of components needed to be automated is a method of reducing dependencies and can be done by identifying the engineering plant architectures. The control system will

be distributed on the component level so to encourage the survivability and robustness of the ship in wartime scenarios.

The approach used by Anteon is to choose a few technologies to demonstrate on a partial mesh of rings topology as a proof of concept. They are attacking the problem by starting small and adding more automated capabilities when there is more knowledge about the design process and functionality. Anteon has chosen the Controls/resource management systems, distributed intelligence/HMI, fluid systems and damage control, situational awareness, equipment health monitoring, network fragment healing, electrical systems, machinery systems, prime movers and propulsion systems [Dunnington et al., 2003]. For the ASDL robust design environment, the characteristics of these subsystems will need to be known. These specifications and characteristics will be provided by the partners in the IRIS project which will specify sending rates, required bandwidth and other network specifications. Until these characteristics are available, assumptions will have to be made to account for the missing system information. In a network simulation, random generation of information can be created based on some general knowledge of the systems acquired by literature searches. For the network portion, this knowledge will be assumed to be known at this point for the proof of concept for a network simulation utilizing the footwork of the Anteon Corporation as a starting point.

A dependable network needs to provide reliable communications in the presence of noise within a reasonable time delay. This is true for any type of communication network, but especially for an onboard ship concept. Because of the close proximity of components, interaction between the components is more prevalent than on a network that allows for more distance between networks. The network will be subjected to malfunctioning nodes because of standard network glitches and the added risk of ship attacks. The network for the IEP project, as specified by the Anteon report, will be required to account for malfunctioning nodes. This network must not interrupt any network traffic due to a single or multiple point media failure. Messages being sent across the network must be received at acceptable latencies and the failures must be able to be detected and located in real-time. Redundant path network architecture will aid in accomplishing these challenges. This is accomplished by a topology such as a mesh.

When developing a network simulation, some specifications and some alternative of settings are to be known. For the preliminary design of a communication network and for this tradeoff study, the baseline configuration will take into consideration the specifications from documentation of the IEP, Smartship and Anteon Cooperation projects. The rest of this section will describe these design decisions and create a list of alternatives and settings for a baseline design and a few alternatives to aid in evaluating communication software and to begin designing the communication network.

Generally, the network onboard the IRIS design ship will be required to have high bandwidth, high availability, low whole life cost, low latency, high determinism, high ruggedness, low vulnerability and support for multiple qualities of service. This will enable the various systems such as navigation, communications, machinery control and surveillance, damage surveillance and control and the management of information systems to simultaneously perform the necessary actions to accomplish the goals of the IRIS designed ship [Parker and Bettencourt, 2003]. These properties are important, so what network specifications will meet these network characteristics?

Beginning with a general specification, what will be the baseline network topology? For the ability to use distributed component-level systems, a mesh of rings topology proves to be useful.

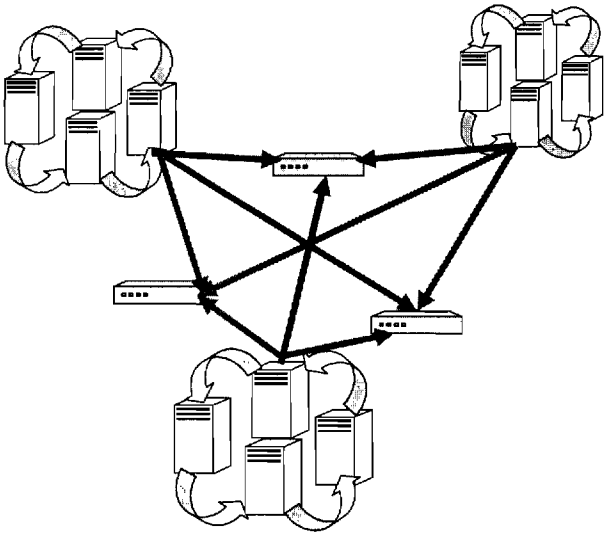


Figure 21: Full Mesh of Rings Topology

The network rings could be the subsystem, such as fire detection or sensors in a zonal configuration, which are connected together in a mesh architecture so that information is stored not only in a centralized location but distributed throughout. In the event of a network failure, this would ensure the data is saved. A full mesh of rings requires extra space for the physical connection of the links for thing rings. A solution for this problem reduces the redundancy but conserves space is the partial mesh of rings. An example of this configuration is depicted in Figure 22.

This topology provides the high bandwidth necessary for carrying out the required tasks of the system [Lively, et al, 2003]. Previous projects with interest in control systems for a warship scenario have explored the use of distributed and independent LANS. This topology utilizes the distributed architecture to make knowledge available across the network. Both the distributed LAN and the mesh of rings topologies make knowledge readily available in the event of a point failure; the damage can be assessed and addressed [Rana and Chhabra, 2003].

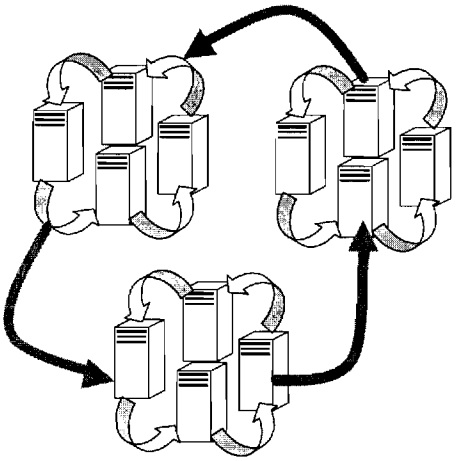


Figure 22: Partial Mesh of Rings Topology

The first step in creating the IRIS design environment is to determine a network creation and simulation tool to be used for finding the optimal network configuration and topology. The rest of this paper will discuss the evaluation of network simulation and creation tools. For evaluation of simulation tools, the topology will be reduced to a simpler form keeping in mind the mesh of rings option and the ability for a tradeoff scenario of network topologies will be important to the final IRIS design environment. The following is a notional topology that will utilize 8 nodes connected by a switch. The topology in Figure 23 will be used to evaluate a network tools' ability to model a switch network with a non-wireless physical layer with the tradeoff of the specifications on the remaining 6 network layers variable.

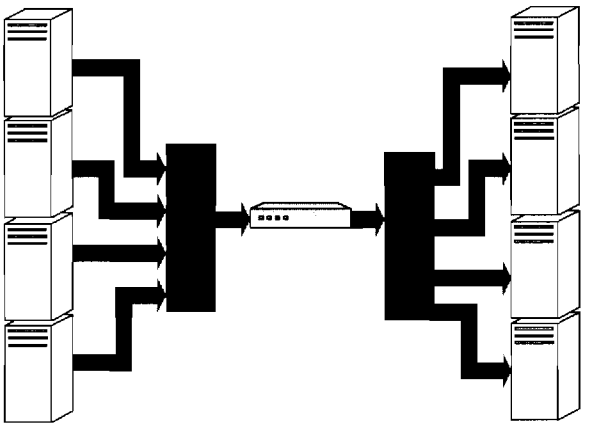


Figure 23: Notional Ethernet topology for use in evaluating software tools

The ability to model wireless topologies will also be important. Not only will the Ethernet type configuration be important, but also a wireless communication configuration. To evaluate a network

simulation tool's ability to model a wireless network topology, the simplified model in Figure 24 will be used [Winkelman, 2005].

Figure 24: Wireless LAN

With any given topology the connection type will be a limiting factor. As described above, the physical layer decision will prove to be enabling or disabling technology. The choices for connection types are vast. Lists of some of the explored physical layer connections for the warship scenario are as listed:

- Ethernet
- Fast Ethernet
- Fiber Channel
- Gigabit Ethernet
- Fibre Distributed Data Interface (FDDI)
- Wireless architectures

Of these technologies, FDDI and gigabit Ethernet are the most mature and suitable for a ship at the current date [Rana and Chhabra, 2003].

Wireless technologies have been explored extensively and have proven to be useful for many subsystems onboard a ship. The personnel health monitoring system is an example of such a technology that would utilize wireless technologies [Gulian et al., 2003]. Consider the situation, if each person on board a ship is to be monitored for position and health concerns but is connected to the observing computer network by a physical line, the mobility of the crew member is limited to the connection length and will have problems interacting with crew members and accomplishing tasks onboard the ship. Thus, this will affect the efficiency of the person. It could be suggested that this function onboard a ship is of utmost importance. With automated decisions of the integrated network the safety of the crew members will be a critical influence of the types of decisions that are made in an emergency situation. The personnel health monitoring system will ensure the safety of the crew members. Wireless technology is the optimal topology.

In summary, the network specifications for the baseline configuration will be:

ATM technology

- Distributed and independent LANS
- Interconnected mesh of rings
- Minimum bandwidth of 20-40 Mbps for non-multimedia data [Rana and Chhabra, 2003]
- Wireless Network: standard wireless protocols IEEE 802.11A&B

Digital device resolution [Louie et al., 2003]:

- 240+x180+
- Frame rate of 15 fps
- 100 Kbps per video stream
- Maximum built in latency: 10-15 seconds

5.3.3.2 Network Creation and Simulation Tool Swarming

With the background of the overall project, networking introduction and the overview of the network applications as applied to the overall IRIS project described, the swarming process of enabling network simulation tools can begin. For a network simulation of the IRIS project, some of the network layers are more vital than others. For instance, the tradeoff of different physical layer models will be a critical to the overall system performance. The delay of information transferred across the network will be critical in the survivability of the overall network. This section will begin to discuss the important features to be modeled in the network simulation, some general network specifications that should be achievable as technology requirements and output data that will be collected. An initial viewpoint on the advantages to commercially available simulation tools as compared to creating a simulation tool will be explained. The Navy's priority is to use as many Commercial-Off-The-Shelf (COTS) products as possible with the new ship design efforts, so this will influence the swarming process and assessment of simulation tools [Wilson, 2003; Edwards, 2003; Kothare and Rana, 2003; Davidson and Nguyen, 2003; Michalopoulos, 2003].

To begin the assessment of the criteria of the software tools available for modeling and simulating network scenarios, a list of functions must be made. This was begun in the previous section but is expanded upon here.

Of the seven layers of communication networks, only a few will play an important roll on the overall design of a communication network. Since several of the layers pertain to the creation of packets and details on a binary level, these will not be the main focus of the network simulation and will be assumed. Rather, in the simulation, the packets will be randomly created within the simulation software. The transport layer, presentation layer and session layer are influenced by this assumption. In final studies of

the overall network structure, these aspects of communication networks will need to be examined, but are not as critical in conceptual and preliminary design phases.

The physical layer, as mentioned earlier, will be the enabling technology. In cases with large propagation delays across the network, the response time of the automated system will be impeded. Therefore decisions will be made to minimize propagation delays. The data encoding portion of the physical layer will not be the primary focus of the network, but rather the technology tradeoff and impacts of the various properties of the connection will be a critical in the trade study.

The format of the information simulated in the overall network simulation will not be critical, but the data correction will be important. The data link layer will be able to detect errors in the packets when they are corrupted from interferences in the system. While the network simulation tool will be immune to such interference, it should be able to model the probability that such errors could occur and emulate a solution to the problem. The actual packet encoding will not be a primary concern when simulating a network, but rather the ability to detect which nodes are intended for the specified messages.

The primary focus of the network simulation software will be to analyze different network topologies including the number of nodes, function of nodes, the number of switches and the switching algorithms associated. Along with this, the network properties such as how often the messages are created, sent and received will be important. The network layer will be a primary focus of the communication network study including the tradeoff of state-of-the-art techniques and architectures.

For the second phase of the simulation of the IRIS project will be to integrate the research by the partner research entities and apply their models for the other systems of the IRIS and IEP projects to model how the actual model will function when the systems are linked together. For this reason, the ability to integrate the model with other modeling software to generate messages to be sent across the network and processed will be an added capability to enable a total network simulation.

Along with the requirements specified in the previous section, minimum requirements of specifications that will be modeled in the network include a bandwidth of 10 Mbps, with 100 Mbps of storage and full duplex connections. The average time that a message takes to traverse the network will be important along with the average amount of throughput on the system in a given time frame will be a useful indication of how the network topology is behaving. The bandwidth utilization, count of lost frames and traffic load will be important parameters to determining how the system performs. The network simulation tool will need to be able to measure these responses of the system.

The goal of the network simulation is to utilize the network topology and network specifications to determine the optimal configuration. This study will explore the available simulation tools and determine what will work best for the scenario of the IRIS project.

- Window's based: Since the partner research institutes are using window-based simulation and design tools, it would be ideal to use a uniform operating system platform.
- Have the ability to be integrated into a collaborative engineering environment where inputs and design criteria constraints can be fed into the communication network software tool and be applied to achieve the optimal solution for the changes of the system parameters.
- The output of the simulation tool should be both visual and text based so results can be readily assessed as well as documented.
- The ease of manually and automatically updating the network simulation tool will be important. The ability to quickly and efficiently edit the design will directly affect the final automated design environment functionality and thus be of utmost importance.
- Be able to model the requirements specified and measure useful output parameters for measuring network performance.

Because of the strict standards, the complexity of the network components and the breadth of the information that will be modeled COTS product are preferable. The validation of the code has been proven and accepted by professionals in the industry. If the simulation tool is to be created in house, the development of each of the components will take away from the overall task of simulating network topologies to discover the optimal solution. The use of COTS networking tool reduces the development effort by reducing the tested and proven code and architectures within the code. However, a simulation tool must have the ability to add new components that are not available in the initial library of components since new technologies are being developed continuously. The focus of the network study can be applied to the application's intent while removing the distraction on the stability and architectural issues associated with creating a modeling tool from scratch. With the focus on the in-depth network study, more robust and secure designs can be created through the use and reuse of a refined network simulation tool. The result will be quicker turn around of assessing the influence of different technologies on the network architecture.

The fallback of using COTS communication network tools is the learning curve associated with the potential "black box" simulation tool. In accordance with this, the simulation toolbox may not be inclusive and as the design study is performed, additional options could be desired that are not offered with the chosen communication network software tool as the project advances and becomes more complex. When an application is created from scratch, the ability to add functionality is readily available and can be implemented. However, many software applications on the market encourage the expansion of the modeling tool and provide information of how to accomplish this.

This section evaluates two COTS tools to determine if either of these commercially available tools will meet the requirements of the IRIS and IEP initiatives in hopes to capture the advantages of COTS tools listed and avoid the disadvantages.

Network Simulation Tools

The purpose of the study described in this paper is not only to lay-out the IRIS problem and introduce network concepts, but to also explore the opportunities of creating a robust design environment by the use of network simulation tools. This section will describe the task of researching network simulation tools as well as exploring a few of the feasible options set by the criteria in the previous section.

Table 2: List of Network Simulation Software and Specifications

	Network Software	Operating System	Language	Price
	Berkley NS-2	Unix (FreeBSD, Linux, SunOS, Solaris)	C++	Free
	PDNS	Unix	C++	Parallel and distributed NS
	Bluehoc	Unix (FreeBSD, Linux, SunOS, Solaris)	C++	Free
	REAL 5.0	Unix/Linux		Free?
*	**OpNet**	**Windows**		**Free and navy is a primary user**
	Qualnet	Windows, Solaris, linux	Parsec	2-week free trial
	QualNet WiFi			2 week free trial
*	**GloMoSim**	**Windows**	**C++**	**Free**
	MIMIC			Evaluation copy
	OMNeT++	win32, Unix	C++	Open Source & free for non-profit Customizable
	Desmo-J	MS, Unix, Mac	Java	Open Source - Free - Primarily for education
	C++Sim	SunOS 4, Solaris 2, HP9000s300 & HP9000s700 workstations using HPUX, Windows 95 and Windows NT (3.5.1 and 4.0), and i386 compatible machines using Linux	C++	Open source & free
	ADEVS	Unix	C++	Open source & free
	Dex (opengl)	Linux		Free, no source
	Swarm	Windows xp		$
	oopm/moose		C++	$
	Simpack		C++	Models physical processes
*	**Simulink/Matlab**	**Windows**		**In the lab**
	Swans		Parsec	
	ACSL extreem	Unix, pc	ACSL Language	we have
	GTNetS	Unix	c++	Free
	ModelNet			Free to academic - unable to download

The communication network community does not have a preferable network software solution for simulating networks. There has been several simulation tools created in government, industry and academia. The ability to model a network is a cost saving exercises and thus is important to all project managers. For this reason, a plethora of software tools have been developed and many are available for free or by retail. Table 2 lists the software options that have attributes that can contribute positively to the IRIS project.

The yellow shading indicates high importance in criteria. The orange is a favorable option but is not optimum for the IRIS project. Based on the simulation tool criteria, it has been narrowed down to the most feasible few. These choices are listed in Table 3. Because of timeframe and from the ability to narrow choices, only two of these were explored at length and are explained in this document. MATLAB™ Simulink is readily available in the lab and is a familiar software tool for all engineering and science fields. Because of the availability and familiarization, this is the first tool to be inspected in reference to the IRIS project. OpNet is a free software tool that is available on windows, which meets the criteria of a simulation tool. For this reason, this software solution will be explored as a comparison to Simulink.

Table 3: Narrowed List of Network Simulation Software and Specifications

Network Software	Operating System	Language	Ease of Use	Price
OpNet	Windows		Easy to Medium	Free and navy is a primary user
Qualnet	Windows, Solaris, linux	Parsec		2-week free trial
GloMoSim	Windows	C++		Free
OMNeT++	win32, Unix	C++	Medium to Hard	Open Source & free for non-profit Customizable
C++Sim	SunOS 4, Solaris 2, HP9000s300 & HP9000s700 workstations using HPUX, Windows 95 and Windows NT (3.5.1 and 4.0), and i386 compatible machines using Linux	C++	Medium to Hard	Open source & free
Simulink/Matlab	Windows		Easy to Medium	In the lab

5.3.3.3 MATLAB™ Simulink

Simulink utilizes a graphical interface to create block diagrams of circuits and systems for model-based simulation and design [Mathworks, 2004]. Component libraries are available for free and are expandable to include specific properties that may not be available in original block set. Simulink uses the ability to utilize hierarchical design for organization of large complex problems. The ability to interface with other simulation programs and incorporate hand-written code increases the functionality markets the simulation

tool to be highly useful for a multidisciplinary design project such as IRIS. Simulink offers the option to run fixed or variable step simulations of time varying systems interactively or through batch simulation.

Mathworks Company has a highly useful website with a plethora of information. What is not found in one of the several manuals available through Mathworks can be posted in a forum and other users in the community will respond with a knowledgeable answer. There is a file exchange were users upload models they have created and make them available for the general Simulink user population [MATLAB Central 2004]. From these models, a starting point can be made on more complex models without recreating complex systems.

Simulink is evaluated with two of the notional topologies noted in previous sections. A communications network with 8 nodes and 1 switch was created and results were obtained about the network performance as well as a network with 2 nodes and communicating wirelessly. Consequently, these examples were found on the file exchange. With these two examples, a good understanding of the usefulness of Simulink could be explored.

Simulation

Ethernet Example

In this example, a network topology with 4 input nodes are connected to a switch and the randomly generated information traverses the network to the destination nodes on the right hand side of Figure 25. This model was found on the file exchange page of the Mathworks website. Ironically, this topology is one of the topologies that were suggested as a preliminary research topology. This model utilizes a hierarchical structure. To create this model, complex block diagrams of digital logic are used to simulate a network function.

Output

The output of this simulation is a representation of the messages as they are being sent from the input nodes to the output. These results can be seen in Figure 26. This graph is a histogram of the bits as they are encoded and sent across the network. The delay in the signals can be seen by matching a shape on the input with the same shape on the output. The contents of this information have been organized in two ways as shown to the right and left of the figure. The method on the left indicates the origin or destination of each message. The representation on the right shows the messages' origin and destination in the order that they have been created. This model has proven that an Ethernet switch configuration can be created and generate results to indicate the performance of the network.

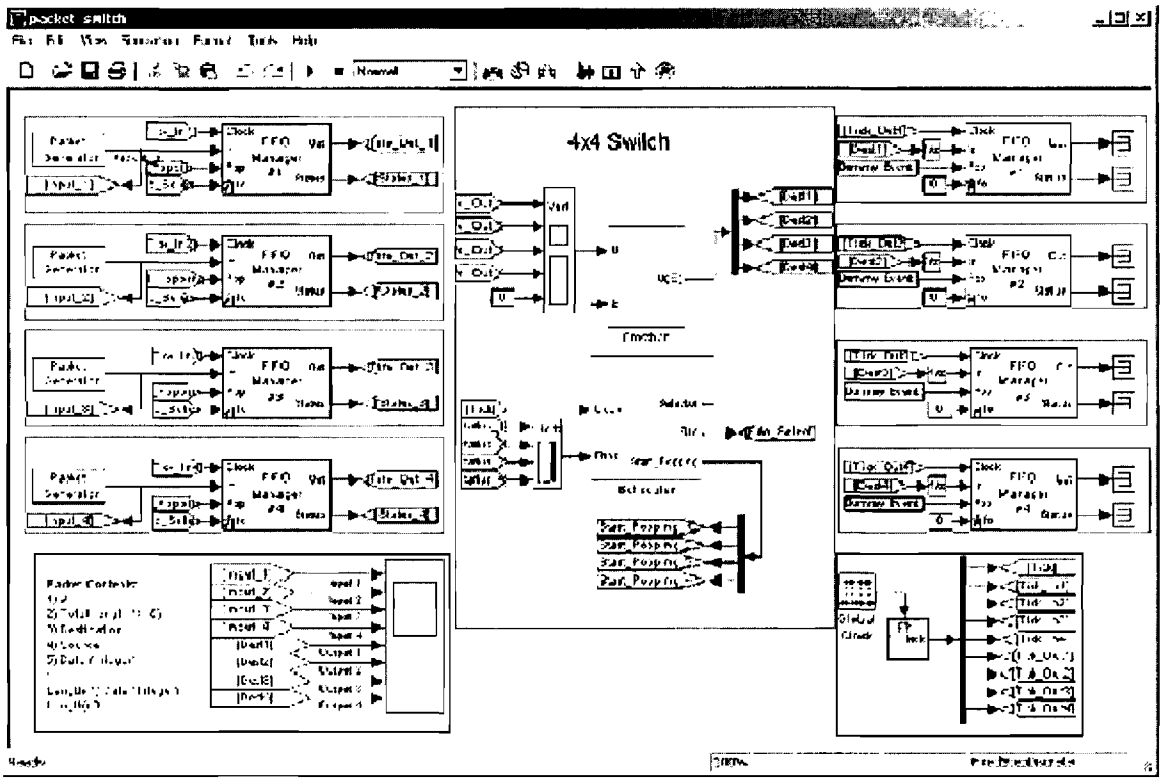


Figure 25: MATLAB™ Simulink Example of an Ethernet Network

Sent to: 2 3 2 1

Sent to: 2 2 1 3

Sent to: 2 4 4 4

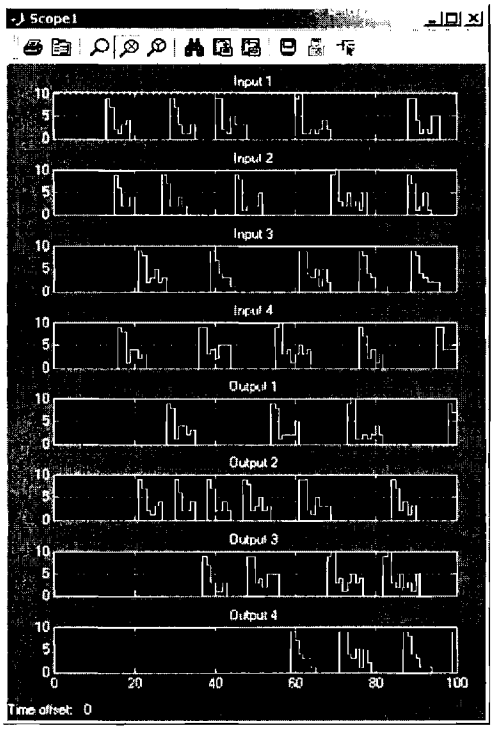
Sent to: 1 3 3 2

Sent from: 4 2 1

Sent from: 1 2 2 3 1 4

Sent from: 1 4 4 2

Sent from: 3 3 3



Send	Receive
1	2
2	2
4	1
3	2
2	2
1	3
4	3
3	4
1	2
2	1
4	3
1	1
3	4
2	3
4	2
3	4
. . .	

Figure 26: Output from the MATLAB™ Simulink Ethernet Example

WiFi Example

Modeling a WiFi network will be important in a tradeoff study for the usefulness of comparing infusions of technologies. Again on the Mathworks file exchange, a model of a simple WiFi network was uploaded to the server and utilized as a proof of concept for this project. In this model, there are 2 nodes communicating with each other via a wireless connection (Figure 27).

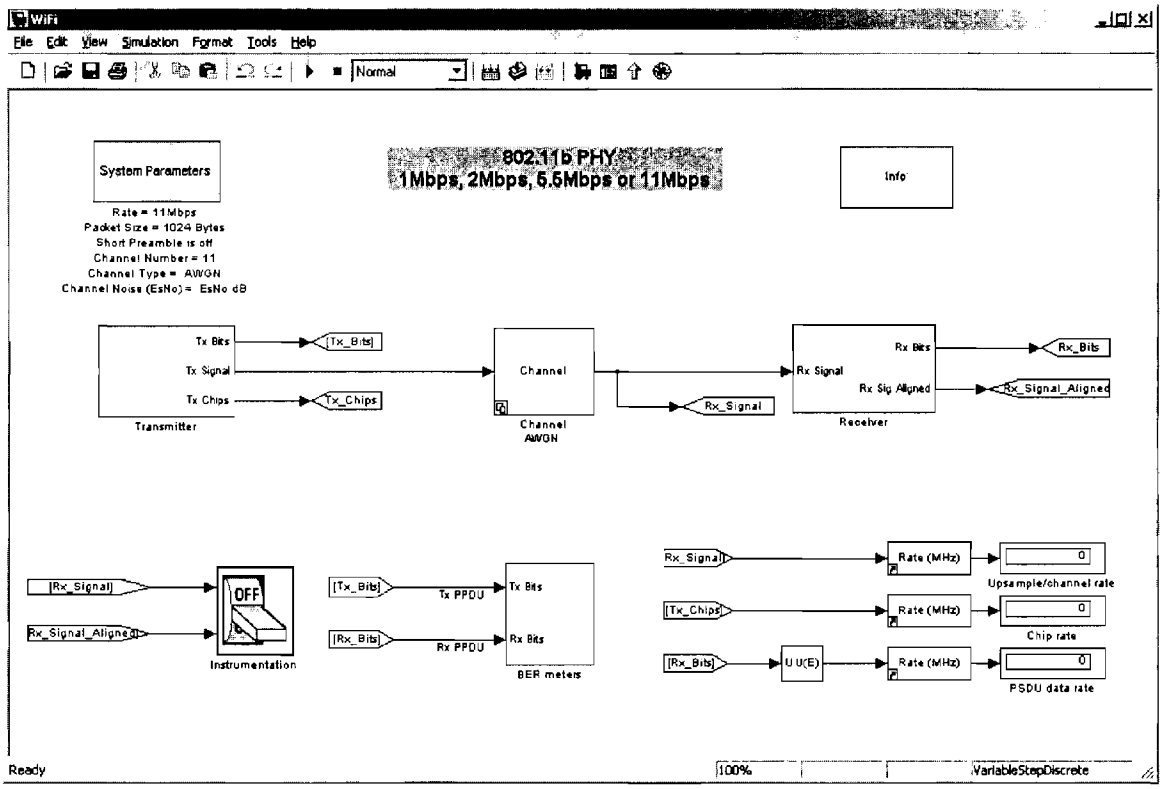


Figure 27: MATLAB™ Simulink WiFi Network Example

This model implements IEEE Std 802.11b – 199 PHY with a choice of mode data rates of 1Mpbs, 2Mbps, 5.5Mbps or 11Mbps [IEEE, 2004]. The packet sizes range from 1-4095 bytes with a short preamble as an option of 2, 5.5 or 11Mbps. The channel number is selected from 1 to 11 and utilizes no channel or an AWGN connection with noise interference as an option. This block diagram utilizes the hierarchical functionality of Simulink to reduce the complexity. Inspecting the BER inverter block, the block diagram in Figure 28 is revealed.

Inspecting the top level view of the network, a switch for turning on and off the instrumentation is seen. By turning this switch to the on position and depressing the play button, the results of the network are displayed in the form of graphs.

Again, for this model, knowledge about details of wireless and digital technology is required to creating and expanding this network scenario. This model does include several attributes that will be

useful in the IRIS environment. The switch will allow for portions of the network to be turned on and off to simulate the impact on the network. Also, this model has the built in capability to vary the input parameters such as data rates. This will be an important feature when solving for an optimal configuration.

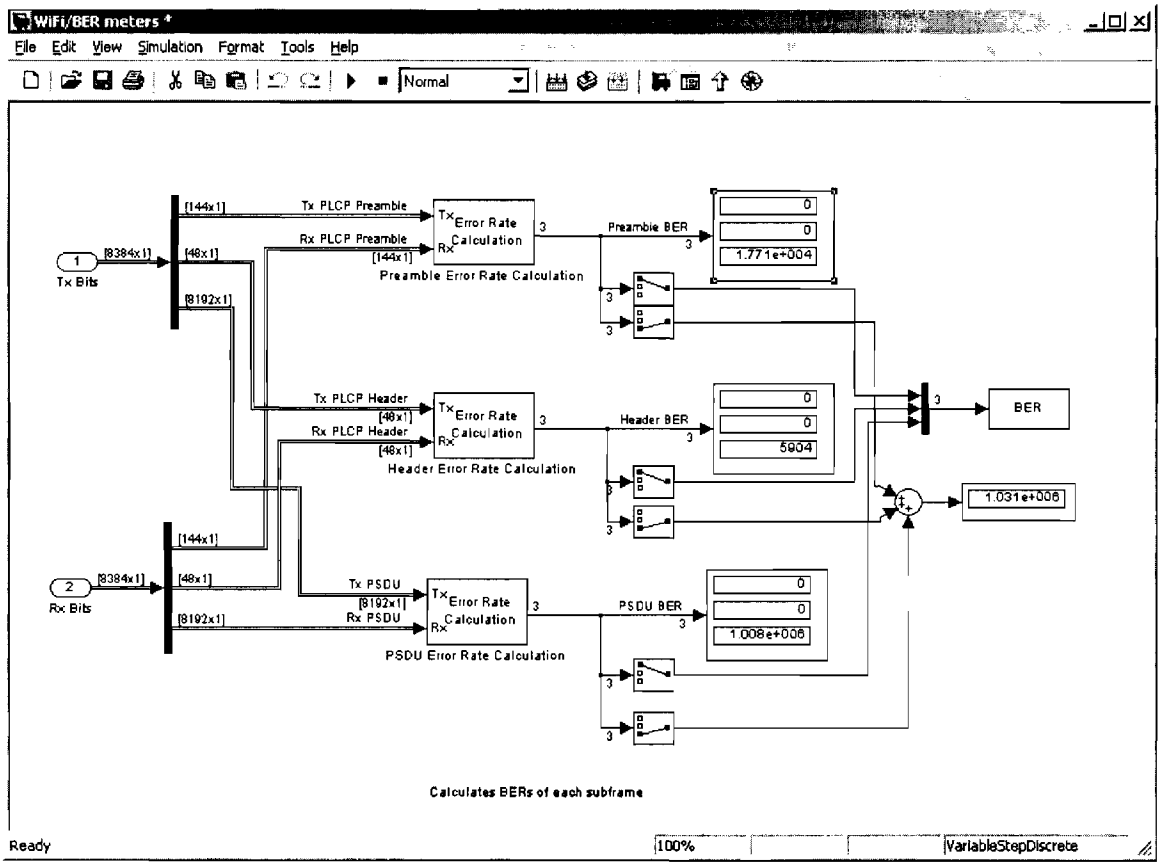


Figure 28: BER Inverter Hierarchical Block for WiFi Network

Output

When this scenario is run with the enabling switch in the on position, output is generated in the form of frequency of signals and error on the system. The outputs graphs generated by the file exchange version depict three graphs. The first depicts the decibel and frequency levels of the signals traversing over the network (Figure 30). The second is an eye diagram which help to visualize and accurately measure timing jitter across the system for both in-phase and quadrature amplitudes (Figure 31), and finally, the overlapping of the amplitudes create another variation of the eye graph to analyze the error on the system (Figure 32). The details of the output are not explicitly important for this stage of the project. This paper focuses on the proof of concept that simulation models could be used for the scope of the IRIS project.

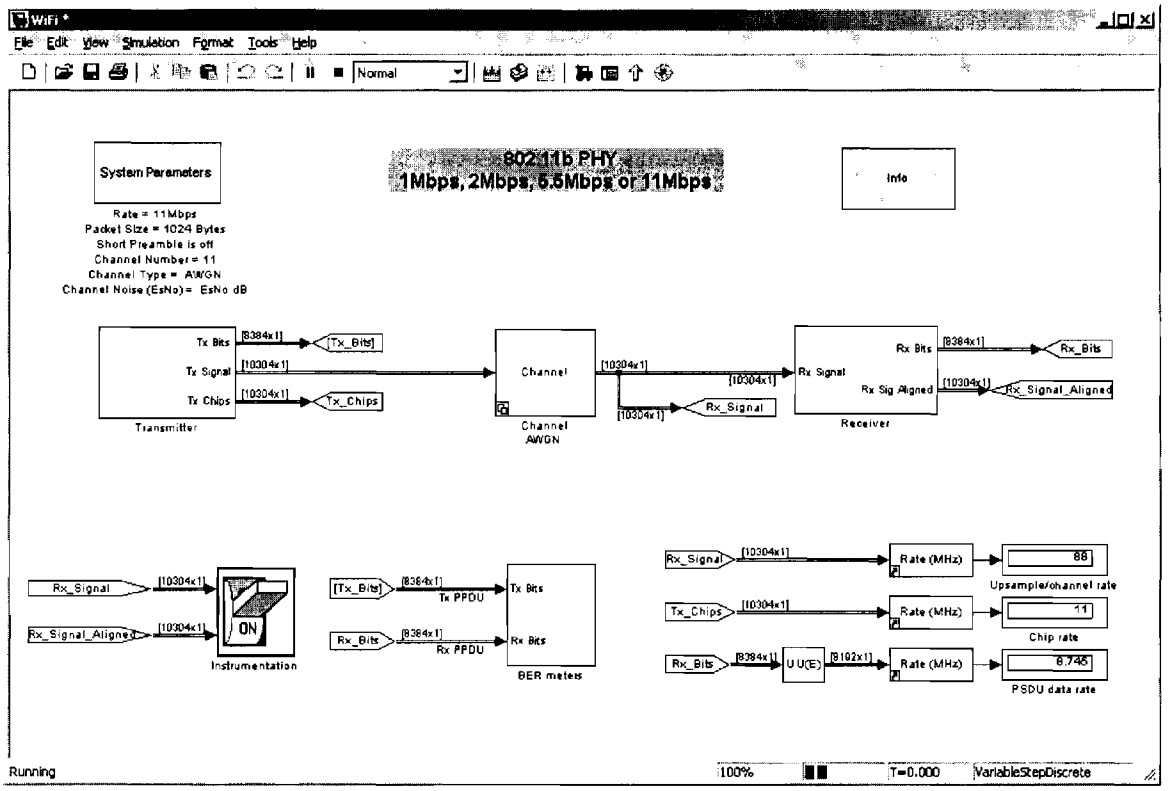


Figure 29: MATLAB™ Simulink WiFi Network Example (Instrumentation ON)

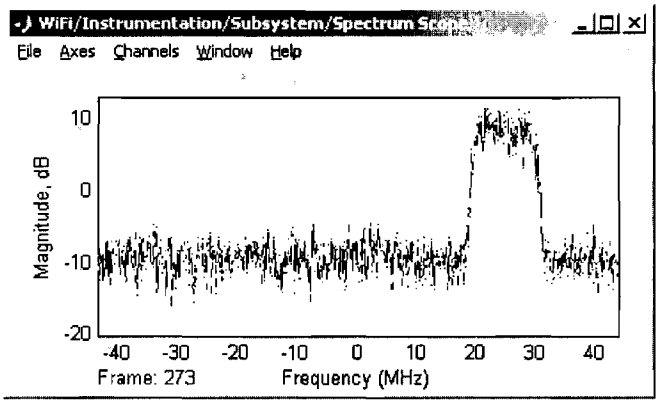


Figure 30: WiFi Output: Magnitude vs. Frequency Plot

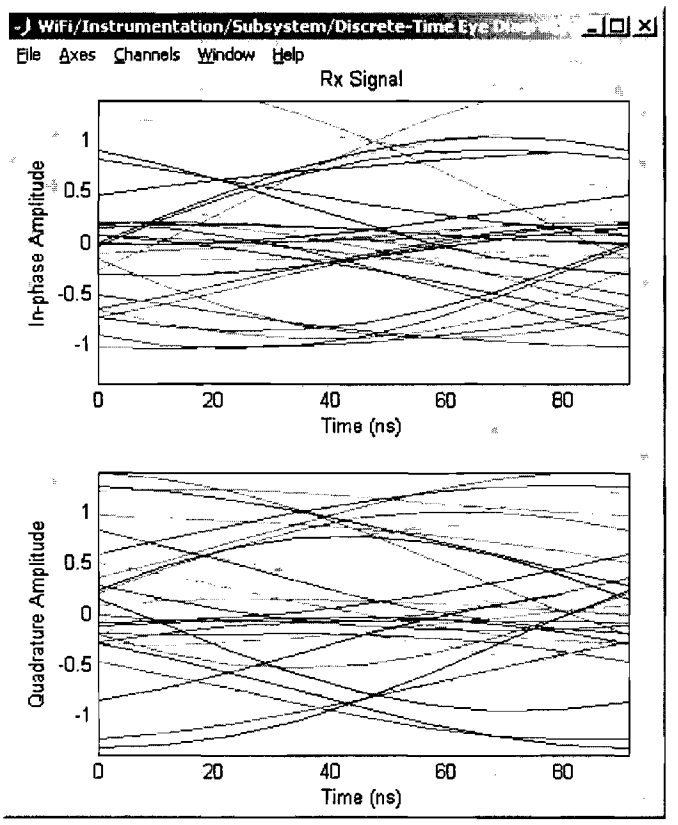


Figure 31: WiFi Discrete-Time Eye Diagram

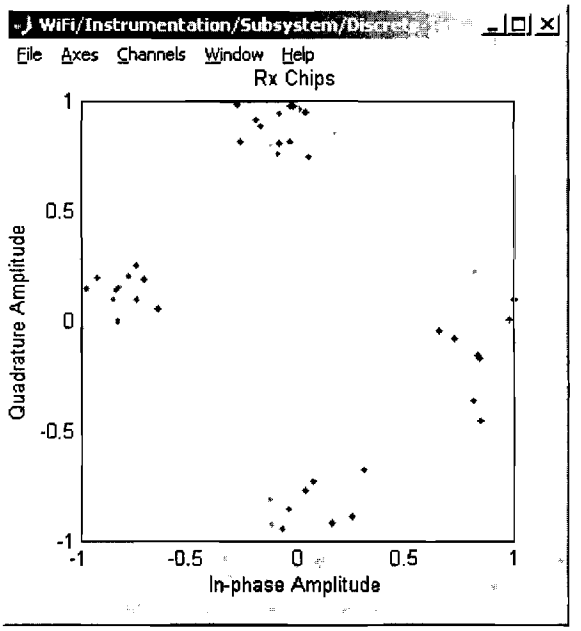


Figure 32: Quadrature vs. In-Phase Amplitude Plot

MATLAB™ Simulink Summary

Using MATLAB™ Simulink as a simulation tool has proven to be useful. Two scenarios were created and tested for validity for the IRIS project by utilizing models that were available on the Mathworks website. There is a vast amount of documentation for MATLAB™ Simulink not only on the World Wide Web, but also in forms of books, manuals, and from users of Simulink. The learning curve associated with creating a general new Simulink model is fairly quick in most cases. However, to create most levels of communications networks, a good amount of detailed knowledge about communications networks and other fields of electrical engineering are required before creating a model. For conceptual design, the detailed model of the complex system is overly extensive for modeling alternatives. This leads to the main disadvantage for using MATLAB™ Simulink for analyzing tradeoffs of infusion of technologies is that the models have to be altered severely to account for the technology changes. For instance, the hardware required for wire and wireless connections is similar, but in Simulink, the models have to be changed greatly to account for the simple conceptual change in the system. For this reason, other modeling software tools have been explored.

5.3.3.4 OPNET IT-Guru

The OpNet IT-Guru program is good for comparing the "what if" tradeoffs of different scenarios including different choices of connections, topologies additions and removals of components and many other variations of the network [OpNet, 2004]. The simulation tool is simple to learn and results are obtained quickly. Additionally, results can be displayed visually or exported numerically to an excel file for further data format. Documentation for the student version that was used for this assessment exists in the form of online forums and tutorials. For the professional version with additional features, there is more extensive documentation available.

OpNet has the ability to model system of systems using a hierarchy as depicted in Figure 33. The top picture shows the system of system point of view. The individual nodes can be focused on to view the network architecture on and node details can be viewed, as shown in the second and third pictures from the top. OpNet IT-Guru has the ability to model state and process models as shown in the second to bottom picture. Finally there is an open source code so that the model can be edited and expanded using programming skills.

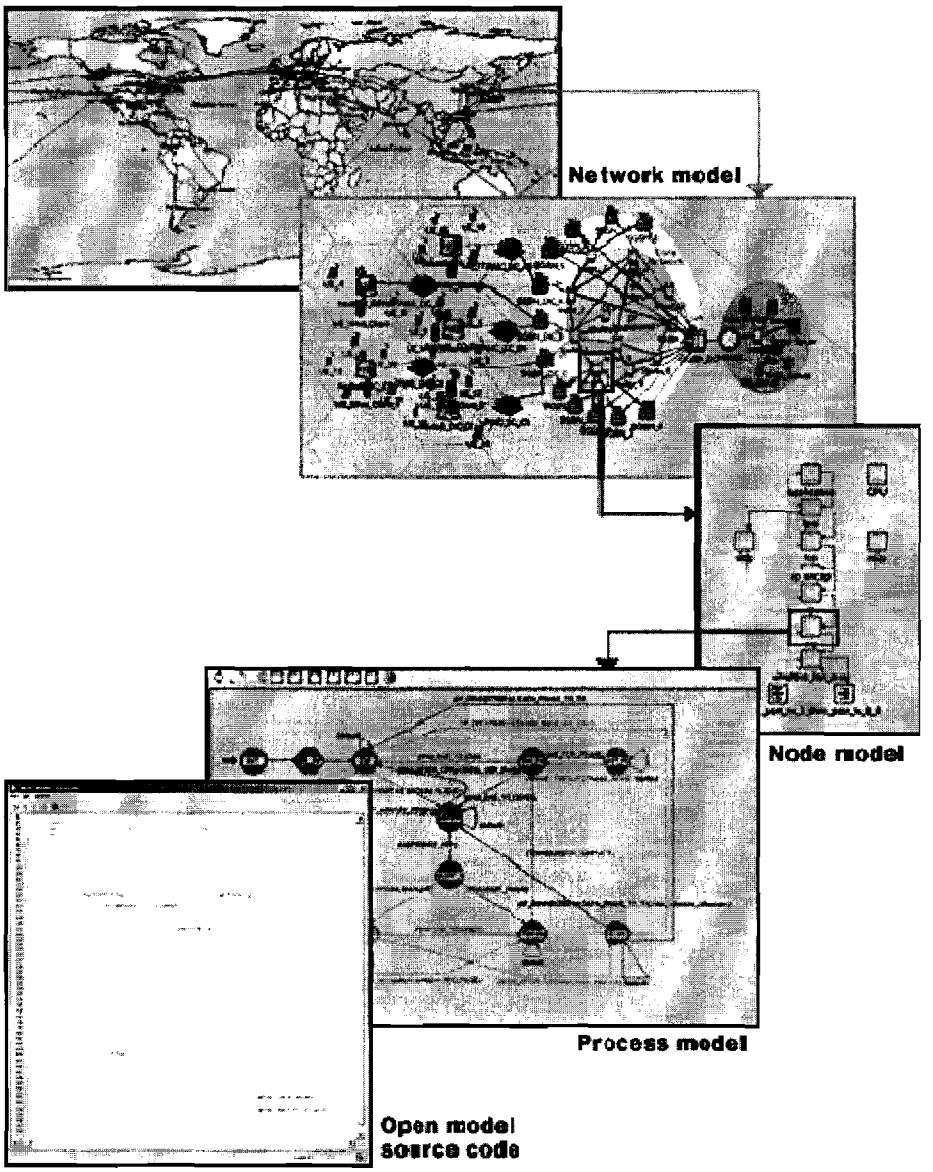


Figure 33: Generalized Functionality of OpNet IT-Guru

OpNet IT-Guru has a suggested workflow methodology to create accurate models of networking systems quickly and easily. First step is to create a network model including the project and scenarios. Then, statistics to be obtained are chosen which will be gathered during the simulation phase. Then the results can be viewed and assessed. These steps will be discussed below.

The first step is to create network model is to create the project and scenarios. A project, in the context of IT-Guru, is a group of related scenarios that explore a different aspect of the network and can contain multiple scenarios. These different scenarios can be run to obtain results for a variety of settings for attributes of the network. From this, the scenario that meets the goals of the overall objective can be identified. When creating the scenario, there are four things that have to be considered: the initial

topology, scale and size of the network, background map of the network and associated object palette (or devices that will be included in the network). The objects on the palette include nodes and links. This program has the ability to quickly create network with a feature called "rapid configuration". In this process, some simple characteristics of the network are chosen and the computer automatically creates a general topology. There are built in protocols that are standard in the networking community. These protocols have been validated by the creators of this project and time can be saved by implementing these features into models that are being created.

The next step is to choose the statistics that are of importance in the tradeoff studies for the various scenarios. There is a plethora of statistics associated with each node as well as for the overall network which are listed in the appendix. For certain cases, some of these statistics will prove to be of little use for the desired comparative data and indicating the performance of the network in a given scenario. Examples of the questions that these statistics will be able to answer are:

Will the server be able to handle the additional load of a second network?

Will the total delay across the network be acceptable once a second network is installed?

The third step in the process of the tradeoff study is to run the simulations. For this process, the user indicates the preferences of the simulation time and process models so that the accurate data can be obtained for comparison. These preferences include attributes such as simulation time and time increments. Running the simulation is as easy as one click of a button. There is a graph that indicates the process of the simulation including the runtime and the average simulation time. Once the simulation is finished, the statistics of the simulation time are displayed and results can then be analyzed upon completion of the simulation.

When the simulation window indicates that the simulation is finished, the results can be viewed and analyzed. The results can be viewed graphically with a simple click of the mouse or a button on the taskbar. The results of one simulation can be viewed or an overlaid graph of multiple simulations. The overlaid simulation graphs are useful in assessing the differences and advantages of the various scenarios. An added, useful feature of the graphical data outlay is the ability to change the type of graph. Some of the choices of graphical display include logarithmic, time average (which is useful in viewing network steady state), average and several other choices of displays.

These results are very easily exported to a spreadsheet for data manipulation. Additionally, models have the capability to set "demand Objects" that will be characterized by objects that has continuous streaming between 2 nodes. This will be important for modeling applications such as video feed.

Simulations

The features of the network simulation software were explored and scenarios were created. This section shows the results of this study including screen shots and plots that were created. Because the software tool was simple to use, a more complex model was created than that with MATLAB™.

Ethernet example

For the first example, Figure 34 shows a screen shot of the OpNet IT-Guru screen where a simple network topology was created. There are 40 nodes connected to a 3Com switch and a server computer. The connections are standard Ethernet line. This model was crated using the built in wizard that allows for a user to quickly generate a scenario and set network attributes.

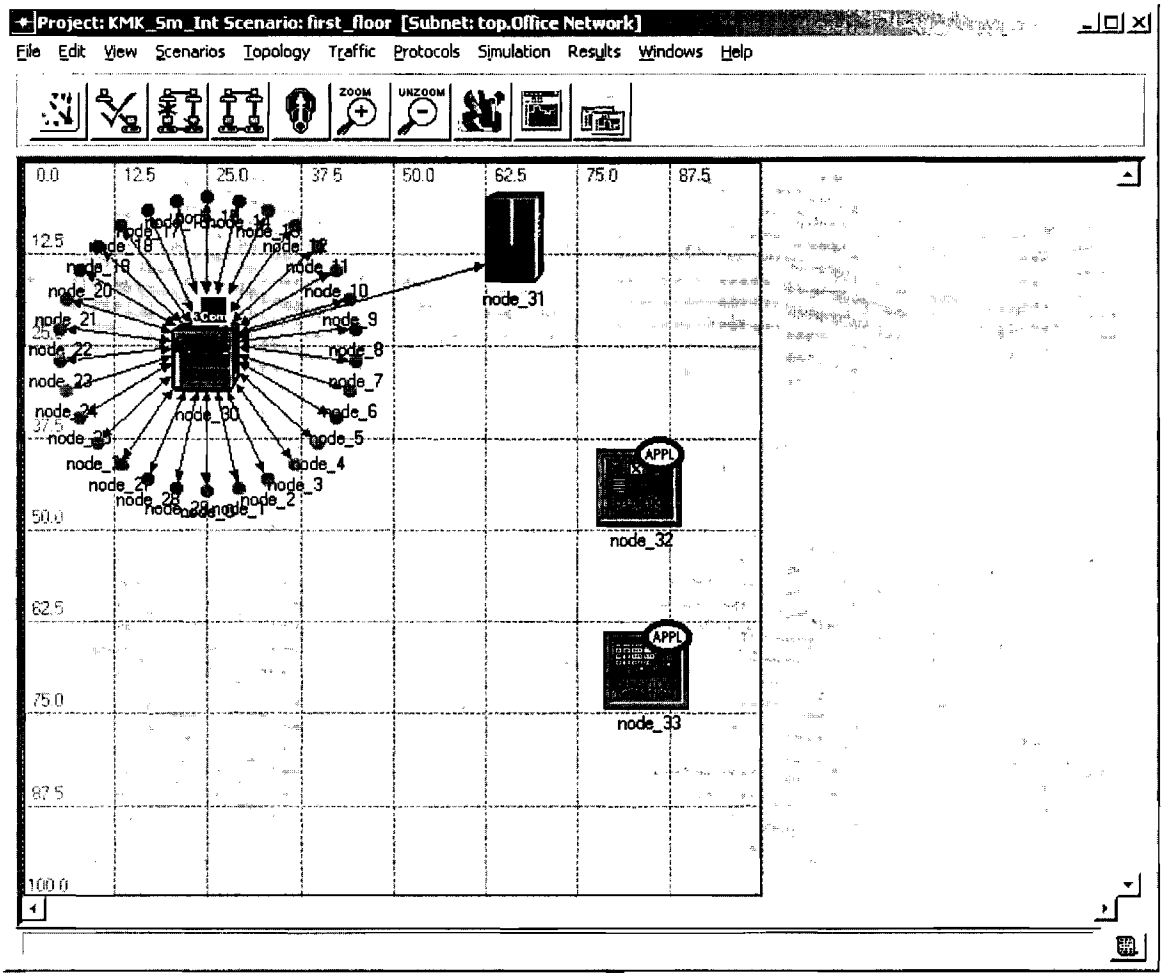


Figure 34: Ethernet Example for OpNet IT-Guru

With the model created, the choice to collect data of Ethernet delay, Ethernet load and point-to-point throughput was selected and are depicted in Figure 35 graphs A and B. These three parameters are

important when assessing the performance of a network. Another scenario will need to be created in order to determine the relative performance.

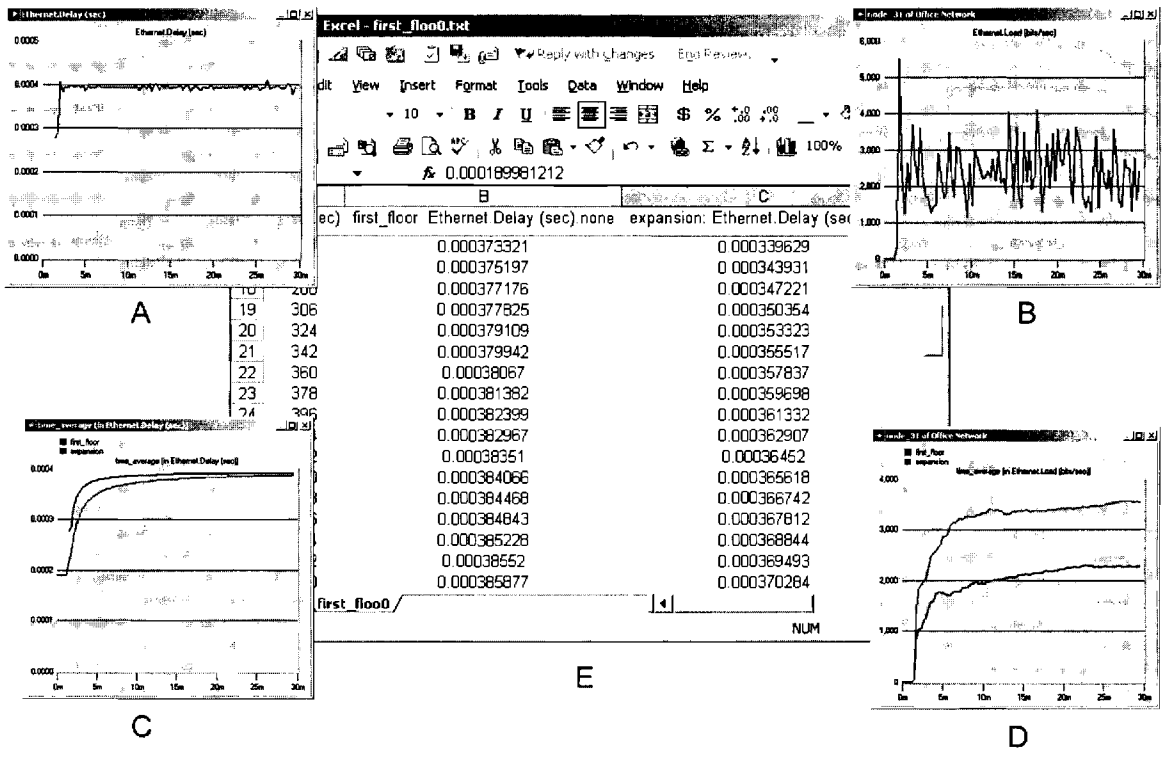


Figure 35: OpNet IT-Guru Output

The new scenario can be as similar to the previous network scenario as changing one attribute of the network topology and as drastic as a complete change in network topology. This will be used to determine the change in network performance between the network scenarios. Tradeoff studies can be explored by creating multiple scenarios. Most properties of the components can be altered to allow for additional tradeoff studies. Figure 36 depicts the same network as in Figure 34 but with an added sub network of 15 nodes connected to the original network via a switch.

A comparison of the Ethernet load is depicted in charts C and D of Figure 35. Here the original topology is in blue and the new arrangement is in red. There is an increase in network load, but it is still at an acceptable level. In this figure, the ability of changing settings is also shown. Here the graph is shown as a time-average. The previous graphs are depicted in the "As Is" selection. The ability to change the plot parameters is important since changing the view can depict different views of the data obtained to arrive at different conclusions or to confirm conclusions. All of these graphs can be saved in a remote location and utilized in further analysis outside of the OpNet IT-Guru environment as depicted in chart E in the center of Figure 35.

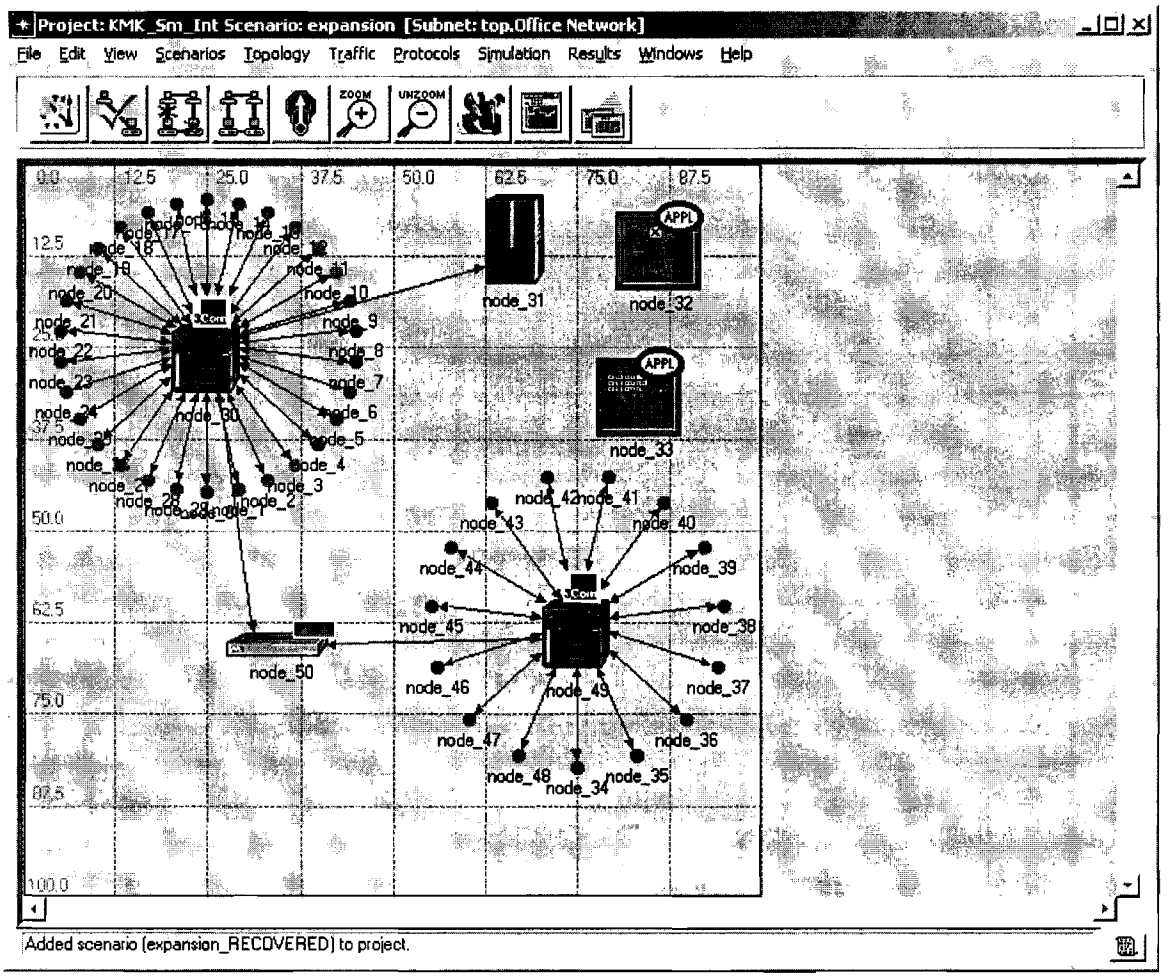


Figure 36: Ethernet Example for OpNet IT-Guru with Two Networks of Rings

OpNet IT-Guru Summary

OpNet IT-Guru as a simulation tool for evaluating tradeoff scenarios for communication network scenarios has proven to be very useful. The software is easy to learn with several examples and tutorials. The software version used for this evaluation was the Academic Edition of the software, which not only has limited functionality, but also has limited documentation. The Research and Commercial Editions have more functionality and much more documentation available. The applications can be set at each node and view the impact of those applications for each node. This will be useful in the iris project for modeling nodes that act as sensors which will have predictable sending sequence as well as other nodes with less predictable characteristics such as control nodes.

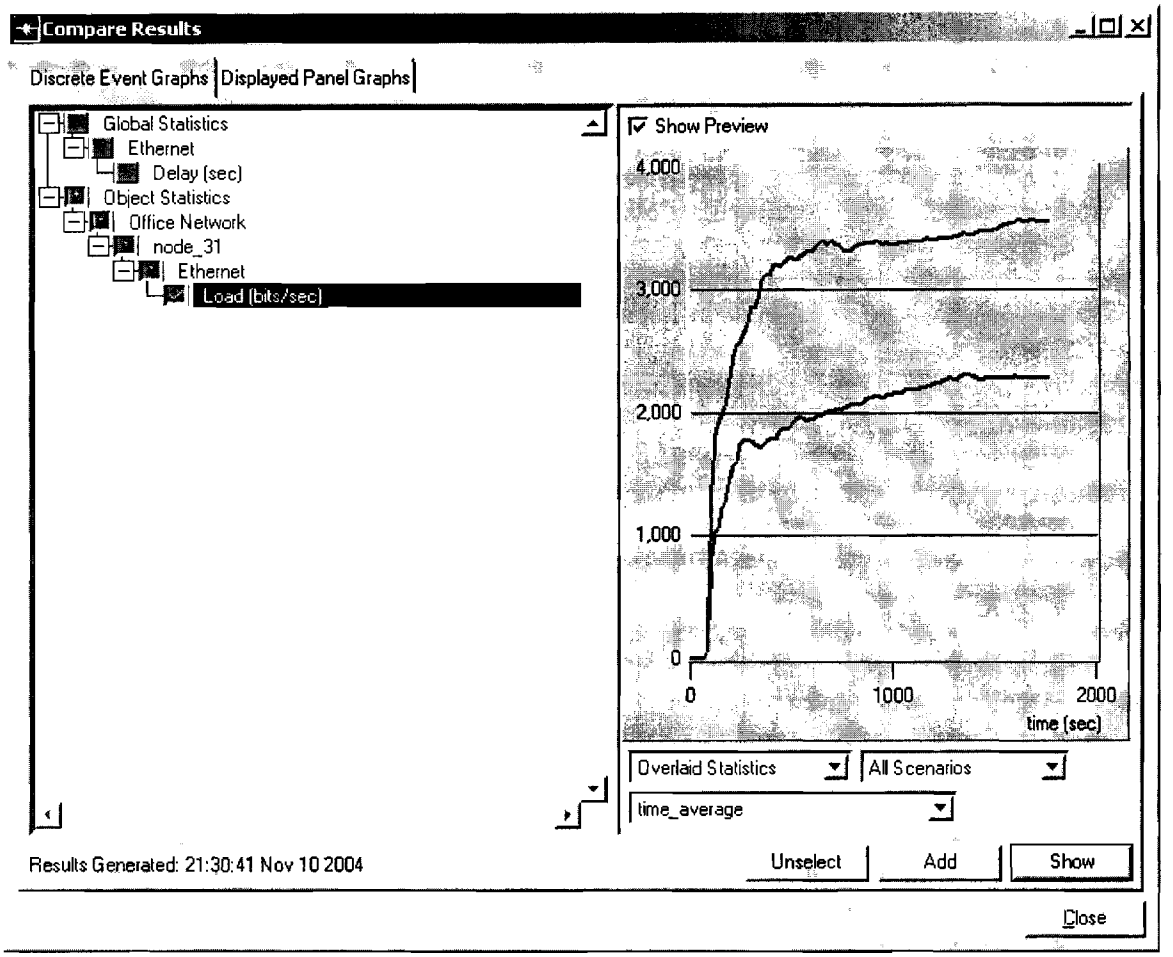


Figure 37: OpNet IT-Guru Comparison of Ethernet Load

From the documentation on the World Wide Web, there are several differences indicated between the student and commercial version of the software [OpNet IT-Guru Manual, 2004]. The Academic Edition of IT Guru is specifically restricted from more extensive export capabilities of the commercial products. The commercial version has the added function to create web reports. This functionality will allow for reports to be created from the graph panels with a click of the button, the report can be written. The Academic Edition is limited to a max of 20 nodes connected to 2 devices, although in this example this was avoided. If this is to be used for the IRIS project, more extensive modeling capabilities is required. The Academic Edition also has a limitation of the number of operations that can be performed (50M operations). This will also be al limiting factor for the IRIS project considering the scope of the project. Other limitations in the academic version is the ACE built in module (which was utilized in the example for a simple case) and wireless functionality. The projects that one can create with academic version are limited to a certain number of multi-port devices. This limits the types of studies that can be performed with routing elements to a reasonable number for educational purposes but does not allow enterprise-level

studies of a commercial product. For a complex network that will be modeled in the IRIS project, these functionalities are required for accurate assessment of tradeoff scenarios as well as optimal design. With the commercial version, add-on modules are available to add functionality to the network simulator. The Research Edition is available for free and has the same capabilities as the Commercial Edition but with stipulations of use such as providing updates to the company concerning the research that is being performed with the tool and creating a website for others to view.

Overall, this product has the capabilities available at the detailed level that is necessary for an accurate representation of network design on a conceptual design level and is favorable for the IRIS project. The use of the commercial version will be useful for the complete IRIS project.

5.3.3.5 Making the Decision

Now that the simulation tools have been evaluated, a decision of the simulation tool of choice has to be made in order to begin modeling a network for a tradeoff study of the IRIS project. The ideal simulation tool was described earlier to be windows based with the ability to be integrated into a collaborative engineering environment with the output from the simulation tool able to be captured both visually as well as in text form. Of the network simulation tools evaluated, the OpNet IT-Guru is the best match to the criteria. IT-Guru is easy to learn and quickly creates results. The built in alternatives of technologies is integral for the IRIS project. For these reasons and those presented in the previous sections, OpNet IT-Guru is the simulation software of choice.

5.4 Control

The interest is to have control as an independent variable, but the path to achieving that is not clear. It is crucial that the behavior of the controls in a time-domain simulation is accurately modeled, and at the same time allow for the capability of studying human-in-the-loop control and the dynamics between the three, the physical environment, the automation systems and the human operators. Figure 38 is a depiction of the hierarchy developed for the automation system of the IEP, containing low level reactive controllers and more deliberate higher level controllers that operate at a more abstract level.

The requirement that the IEP degrade gracefully, meaning that failures will not cascade through the system and damage be contained within a minimum number of low-priority components, leads to the need for a distributed control architecture. The need to have a modular yet robust architecture that permits human-in-the-loop, demands the use of hierarchical control. IRIS can take advantage of the development of autonomous, active controls in an integrated system to create a distributed, intelligent control system that can sense, assess, and react to any anomaly introduced into the ship system. IRIS seeks to create an integrated system with the ability to autonomously control each of these aspects under any condition by employing active controls.

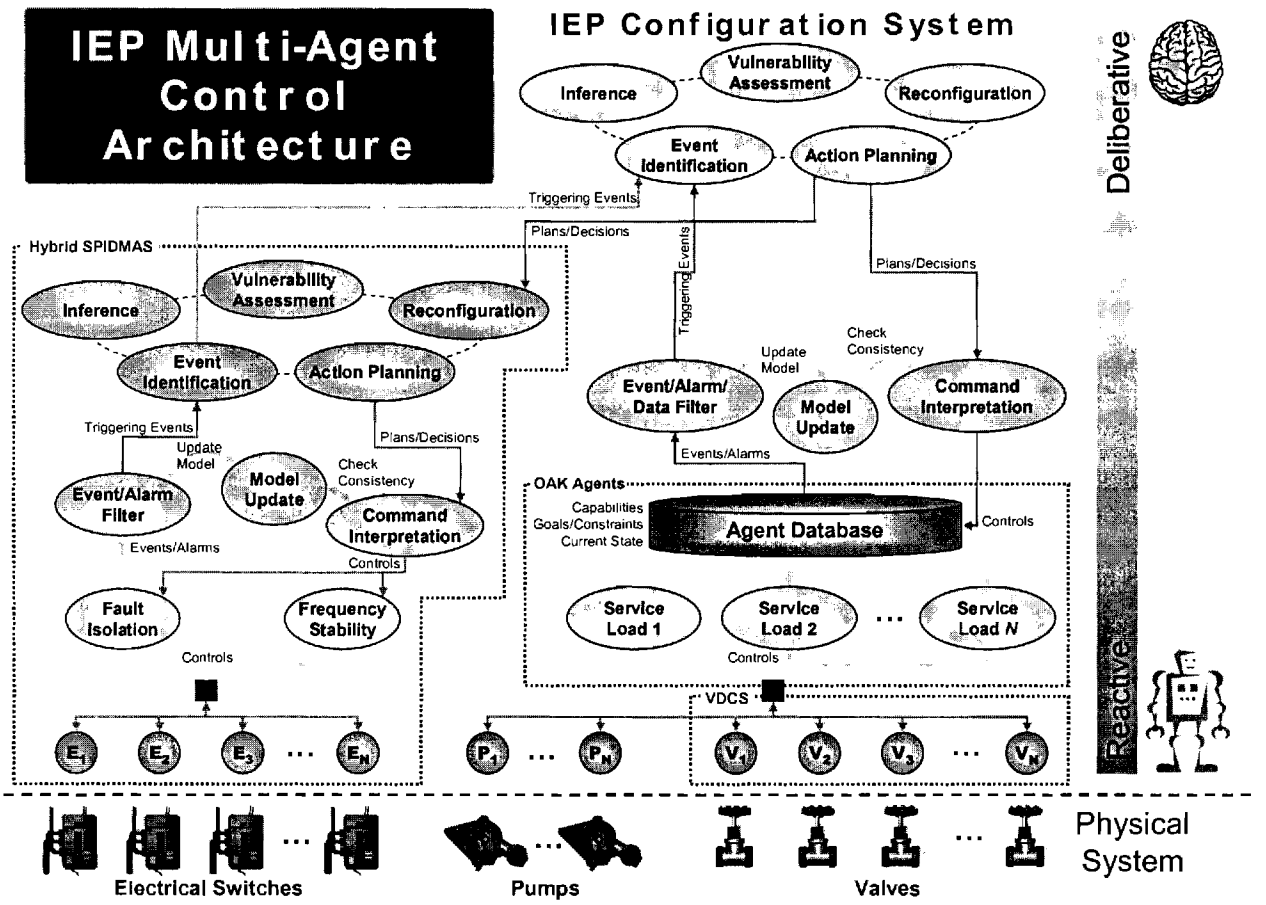


Figure 38: Notional hierarchical distributed control architecture for the Integrated Engineering Plant

5.4.1 Agent-based Control

In the recent decades, there has been a strong interest in control for large-scale complex systems, especially in autonomous intelligent control methods. Classical control systems based on feedback techniques and central control methods generally cannot manage computational complexity, nonlinearity, significant uncertainty and heterogeneity. Especially for the control methods that attempt to design point-design systems, it is hard to deal with situations that deviate considerably from nominal conditions and may lead to catastrophic results. Moreover, it is almost impossible to predict or list all possible damage scenarios and store the corresponding strategies in memory to reconfigure the system.

For a dynamical system with significant degree of uncertainty, robust control is crucial, especially for weapon and space systems. The usual way to do that is to integrate extra copies of some critical components (block redundancy) or make the system capable of achieving the same goal by using different methods with different components (functional redundancy). Choosing either of these two methods, the control system needs to determine the current configuration and states of the system and change to another configuration automatically. However, it is not practical and affordable to install sensors

everywhere in the system, and the sensors as components may fail over time, so the observations from sensors are incomplete and uncertain. Using one controller for the whole system would make the controller design too complicated and its reliability would be low. Moreover, techniques for large-scale complex systems are being developed and upgraded more rapidly every year, which means more components and/or data have to be added. In order to increase the affordability for the controller system, the control system needs to be easy to modify, adapt and expand when parts of the system are changing or more data/components are available.

Hierarchical multi-agent based control methods are the focus in recent decades to solve such problems mentioned above. Hierarchical multi-agent based control is implemented by decomposing a complex system into smaller subsystem; each subsystem is treated as a relative isolated part; all of the parts work interactively by their interfaces (inputs and outputs). Hierarchical multi-agent based control systems have many advantages in controlling large-scale complex systems.

- It divides a big task into smaller ones, which makes it easier to implement many modern control methods such as neural nets control, fuzzy logic control etc to each part
- It separates the controller from the physical model and the controller is designed as software which can handle computational complexity
- It makes the system more robust for uncertain and dynamical environments
- It makes the system more flexible and adaptable

5.4.1.1 Introduction

Agent-based control is a framework of modern control. It is always used to handle complexity of a system in a hierarchical form which is natural and effective. Many different control methods, such as neural nets control, fuzzy logic control, adaptive control, etc., can be used for each agent or used to coordinate the actions between the agents.

An agent is an encapsulated computer system that is situated in some environment and can act flexibly and autonomously in that environment to meet its design objectives" [Wooldridge, 2000]. To be more specific, an agent always has several characteristics as following [Jennings and Bussmann, 2003].

- An agent is a clearly identifiable problem-solving entity with well-defined boundaries and interfaces, which means an agent is relatively isolated and intelligent
- An agent is always situated in a particular environment over which they have partial control and observability, which means the whole system, can not be completely controlled by one agent. Each agent has local control capability and the whole system behaviors are the results of the coordination of many agents

- An agent is designed to fulfill a specific role: has a particular objective to achieve, so an agent is not a random object, it has its particular goal to achieve which may be fixed or changed according to its environment
- An agent can automatically control over both its internal states and its own behavior according to its environments. An agent is a smart entity and can adjust its own actions

An agent is capable of exhibiting flexible problem-solving behavior in pursuit of their own design objectives, being both reactive and proactive. Reactive means it is able to respond in a timely fashion to changes that occur in their environment. Proactive means it is able to opportunistically adopt goals and take initiatives.

Since each agent is relatively isolated and small, it is very easy to dissect a large task into several small assignments to a bunch of agents which makes the design much easier, more adaptive and save time if there are many subsystems with similar structures, which is always the case in realistic life. Moreover, it provides an easy way to expand the system with less effort.

5.4.1.2 Establish Agent-Based Control for Large-Scale Complex Systems

A system consists of many agents. Each agent has a clear interface and boundary to interact with other agents, such as what inputs it needs and what outputs it offers; each agent has its own logic to decide the behavior of itself according to its environment which determined by its inputs. Each agent affects the other agents' behaviors by the outputs.

Hierarchic structure is widely accepted in Multi-Agent-Based Control system considering the following five reasons [Jennings and Bussmann, 2003]:

- The relationship between agents are more clear
- Complexity frequently takes the form of a hierarchy, that is, a system composed of interrelated subsystems, each of which is in turn hierarchic in structure until the lowest level of elementary subsystem is reached
- The choice of which components in the system are primitive is relatively arbitrary and is defined by design objectives
- Hierarchical systems evolve more quickly than non-hierarchical ones with comparable size, which means it is easier to develop and advance hierarchy system
- It is possible to distinguish between the interactions among subsystems and those within subsystems.

The quantities of the layers for hierarchy depend on the structures of the physical model and the requirements of control systems. For example, researchers at John Hopkins' University used 5 layers for

their Open Autonomy Kernel (OAK), a control agent architecture that is based on a unique combination of model-based reasoning and software agents for ship system control. The structure is shown in Figure 39.

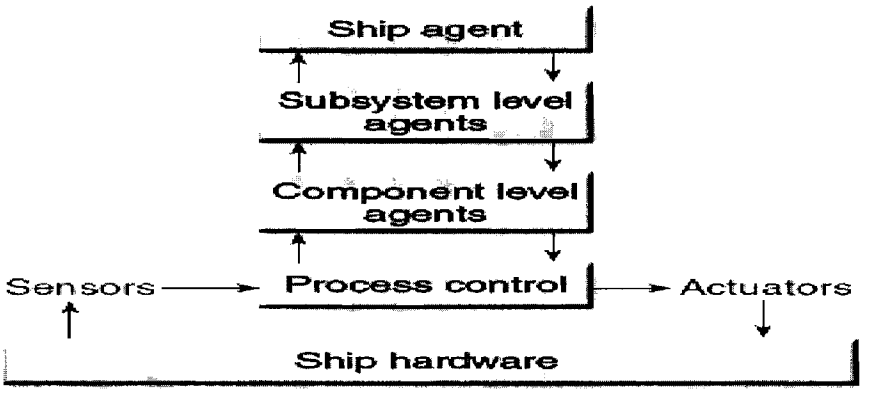


Figure 39: The Hierarchical Control Architecture Implemented in the Open Autonomy Kernel [Maturana et al. 2005]

Researchers at University of Washington designed a three-layer architecture using multi-agent based control for power system in his thesis which is as shown as in Figure 40. The lowest layer is the reactive layer, which is located in every local system and gives immediate response. The middle layer is the coordination layer which uses a knowledge base which triggering events/alarms and update the current model of the power system. The deliberative layer is trying to give a general plan for the whole system according to the information from the middle level agents.

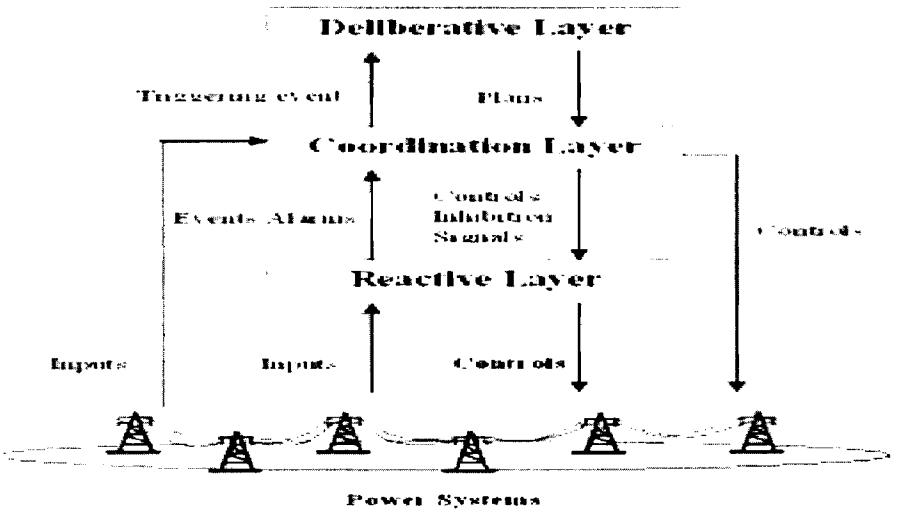


Figure 40: The Hybrid Strategic Power Infrastructure Multi-Agent System [Jung and Liu, 2001]

Is there a general procedure to establish the hierarchy multi-agent based control for a complex system? Firstly, the characteristics of large-scale and complex system need to be addressed. Usually, a

complex system is composed of many parts which are functionally or spatially relative isolated. A complex system is always has the form of hierarchy, which means it can be divided into small subsystems and each subsystem can be divided further and further according to the requirement of the control system for a physical model. The interactions between its components can be addressed by the inputs and outputs for each piece. Usually, a component has scarcity relationships with other components, which means a component does not connected with every other component, it just connected to some of them, which means each component does not have too many inputs and too many outputs then it is easier to establish the relationships among components. A three-step procedure is recommended to build a multi-object oriented control system [Jennings and Bussmann, 2003]: decomposition, abstraction and organization. This procedure can be used to establish a framework for multi-agent based control system.

Step 1: Decomposition

Divide the large complex system into smaller, more manageable pieces and address each piece in a relatively isolated way. A system can be decomposed by its components spatial positions or functions or using both of them simultaneously, which depends on structures of the physical system and the objectives of the control system. Each piece can be addressed relative independent, which makes the logic for each piece much clearer, much easier and close to the actual system.

Step 2: Abstraction

In order to design the internal logic and interface for each piece, the objective and structure for each portion, what it requires to achieve its goal, and what information it supplies must be defined. Depending on its objective and the information available, a relationship of its inputs and outputs can be established which satisfies its simultaneous objectives. Since each portion is relatively small, it is easy to use complex control methods in a relative short time.

Step 3: Organization

Organization is concerned with defining and managing the interrelationship between the various portions. It is the most challenging part in agent-based control design process. However, considering the sparsity of interactions between the agents, it is not so difficult to establish the relationship between various portions. Fortunately, since a complex system is always taking the form of a hierarchy; component in a subsystem would not connect with other components in the other subsystem. So basically, there are just direct relationships between the components in the same subsystem. Subsystem connects with subsystems in the higher level, which makes the establishment of relationships much easier to be organized.

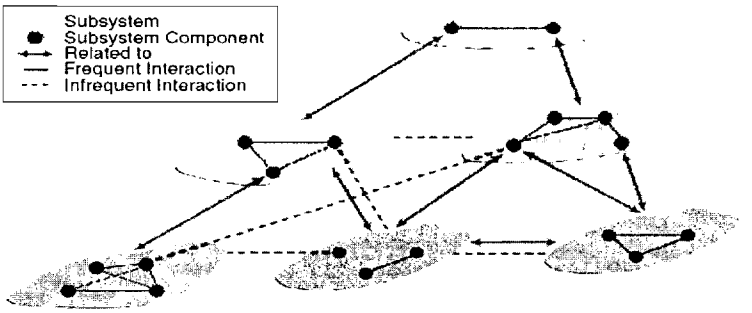


Figure 41: Representation of a Complex System [Jennings and Bussmann, 2003]

5.4.1.3 Develop a Reasoning Engine to Diagnose the System States

Previously, during the establishment of hierarchical multi-agent based control for a complex system, the information from the sensors is determinant and complete which is not practical. There is always significant degree of uncertainty in the information gathered from measurements of sensors, the controller can not use the raw data directly. How to get the more actual and complete information of the system from the observations is the most challenging part for the whole process. Here, a rule-based reasoning engine using partially observable Markov Chain decision process will be introduced.

The partially observable Markov Chain decision process is defined by the quintuple (S, O, C, T, R)

S_{i-1} : Previous finite set of states of the agent being tracked

S_i : Current finite set of states of the agent being tracked

O_i : Current partial observations of the agent being tracked

C_{i-1} : Previous commands to the agent being tracked

C_i : Current commands to the agent being tracked

T : The state transition function maps elements $S_{i-1} \times C_{i-1}$ into discrete probability distribution over S of the agent being tracked

R : The reward function maps S to R which gives the instantaneous reward for the agents being tracked getting into some specific state. R is used for the system to how to choose the command by learning from previous step. It is a learning procedure for the agent.

However, firstly, it is really difficult to get realistic probabilistic distributions between some observations and the real system states; secondly, the sensors as components may fail and have different lifetime distributions under different operating conditions; in order to get the distribution of a sensor's

lifetime under various conditions, tons of experiments need to be done and many practical data need to be gathered and analyzed in some way which is beyond to this paper and will not be discussed here. In the next section, a hierarchy multi-agent based control with full observability is established and tested for a reduced scale advance demonstration model of Navy Ship Chilled Water System.

5.4.2 Dynamic Decision Making Under Uncertainty

The agent based control technique enables the automation at component level, so the objectives of the subsystem, which is composed of several components, can be successfully fulfilled. Since various subsystems work together to achieve the overall goal of the naval ship system, the right tradeoff among the multiple objectives of the subsystems should be done to optimize the objective of the ship system. As a result, proper decisions need to be made from a system's point of view to keep the ship working functionally and effectively. In the ship systems operation, these types of decisions are often made based on the assessment of large amounts of information that describe the state of the system. It has been already discussed on several occasions that the information used to make decisions is usually changing overtime due to the continuously changing situations of the system. This requires the decision makers to be able to make the sequential and interrelated decisions under time constraints. In addition, the decisions are usually completed based on uncertain or incomplete information due to the data availability and the variations in the environment. This fact exacerbates the complexity of the decision making process because it results in the difficulties of perfectly and deterministically reasoning about the effects of the decisions and thus make it hard in determining further decisions. The complexities result in the sequential stochastic decision making process, as shown in Figure 42, which is always a challenge to human decision makers since it is difficult for human beings to manage and organize the time-dependent information and make appropriate decisions based on the probabilistic assessment of the acquired data.

In modern ship operation, more and more emphasis has been given to reducing cost and manning workload, and increasing ship survivability and mission effectiveness. This produces a requirement that the large amount of time-varying information needs to be rapidly processed and the decisions associated with ship operation should be made autonomously. The IRIS framework is proposed as a possible solution to fulfill this requirement. With the reconfigurable systems, the IRIS designed ship will assess the incoming information and then configure itself into the mode most adequate to deal with the situation at hand by taking the best course of action. This requires the system to possess the capability to make autonomous decisions based on the analysis of the incoming information which is uncertain and changing over time. Therefore, in order to make proper decision and increase the system's effectiveness, an advanced decision making strategy is needed to make autonomous decisions while capturing the system's dynamic characteristics and environmental uncertainty.

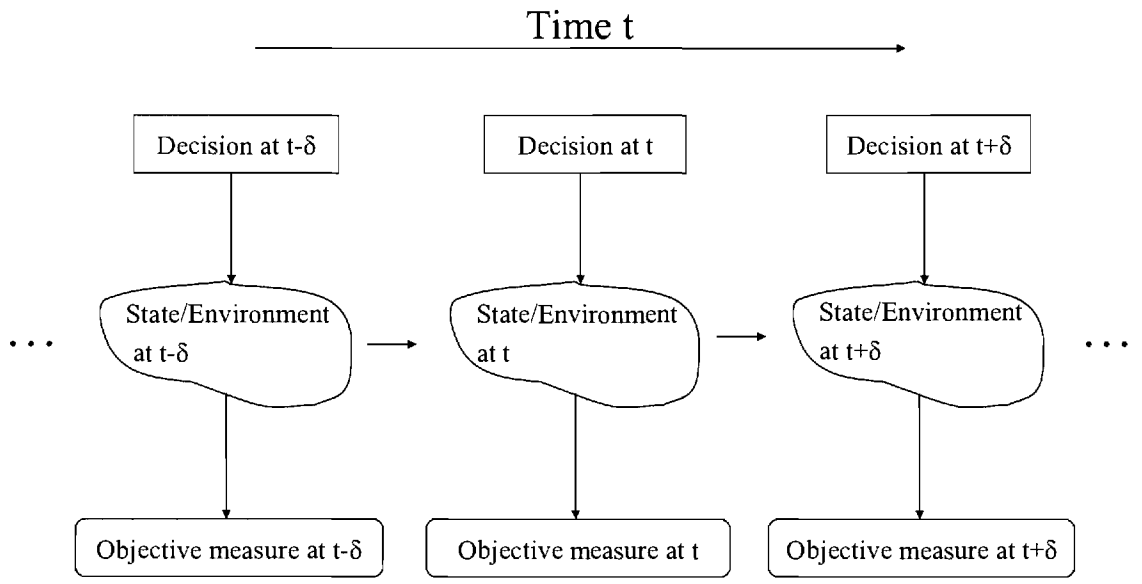


Figure 42: Dynamic Decision Making Under Uncertainty

5.4.2.1 Existing Approaches to DDMUU

Dynamic decision making under uncertainty (DDMUU) is an area where tradeoffs need to be done in an uncertain and real time domain. Many efforts have been made to facilitate the problem solving procedure of these types of decision making problems. As a result, various approaches were proposed, and among these approaches three ones are widely used. They are Dynamic Decision Analysis (DDA), Artificial Intelligence planning (AIP) and Markov Decision Process (MDP).

The decision analysis often employs a model which utilizes the probability theory and utility theory to obtain an expected return or cost, then decide the best course of action to be taken. The decision tree and influence diagrams are two typical analytical formalisms in decision analysis. DDA techniques, such as Markov cycle tree [Hollenberg, 1984] and stochastic tree [Hazen, 1992], are based on the traditional decision analysis models and are capable of representing the stochastic process of the dynamic decision problem. Dynamic decision analysis requires the decision maker have knowledge about the consequences of the decision, such as the effect of each action that may be taken in the decision making process. This information is often uncertain and needs much effort to be discovered. This difficulty prevents dynamic decision analysis from being an appropriate method for decision making under uncertainty.

The AI planning is used to provide a plan that is a fixed sequence of actions to achieve the goals in a dynamic environment [Newell and Simon, 1963]. Modern AI planning takes incomplete and uncertain information into account and is able to generate planning for a stochastic process. AI planning involves the representation of actions, reasoning about the effects of actions, and techniques for efficiently searching the space of possible plans. This approach can handle the uncertain conditions in the dynamic

decision making process, but the use of AI planning requires the developer to identify and handle complex numerical and logical relations. Moreover, the fixed planning is usually not suitable to handle the domain-dependent planning problems because the domain-specific information and knowledge varies with domains and the planning processes are significantly different. These disadvantages make AI planning difficult to apply in practice.

Markov Decision Processes (MDPs), also known as controlled Markov chains, were invented by Howard in 1960 [Howard, 1960]. This approach provides a mathematical framework characterized by a set of states that the system could be, a set of actions that the decision maker has to choose in each state and a transition matrix that represents the probabilities of one state transiting to other states if a certain action is executed in the original state. A reward is earned after a certain action is executed in a specific state. The solution to a MDP is an optimal policy defining which action should be taken for a given state, regardless of prior history. MDP is found to be surprisingly rich in capturing the essence of sequential decision making under uncertainty, and it was successfully applied in many areas, including operations research, control engineering, decision sciences, and so on.

The comparison of the three approaches shows that though the dynamic decision analysis and AI planning have their own advantages in handling the dynamic decision making problem with uncertainty, these two approaches have difficulties in practical application. On the other hand, the Markov decision process has been successfully implemented in many areas and appears to be a promising approach to the DDMUU problems.

5.4.2.2 Markov Decision Process Model

The Markov decision process is an extension of Markov chain, which is a discrete time stochastic process describing the states of a system at successive times. At these time points, the system changes from one state to another or stays in the same state. The changes of state are called transitions, and each transaction occurs according to some probability called transition probability. Markov chain must satisfy the Markov property which states that the transition of the system depends only on the current state, but not on the states in the past.

A Markov decision process is a Markov chain with actions and rewards. The actions are the alternatives that have to be chosen in each state, and the execution of an action will cause the system transits to the next state. After an action is performed in a state, a reward will be earned for this state action pair. The MDP is depicted in Figure 43. The reward of the action-state pair plays a critical role in determining which action should be chosen in each state. Notice that in a MDP the best action taken in a state is not necessary the action resulting in the immediate maximum reward in the state. This is because the action with immediate maximum reward may cause the system to transit to an undesired state in the

future. Therefore, to choose the best action tradeoffs should be made between the immediate rewards and the future gains to yield the best possible solution. This indicates that the goal of the MDP is to maximize the accumulate rewards so that the best course of action can be determined.

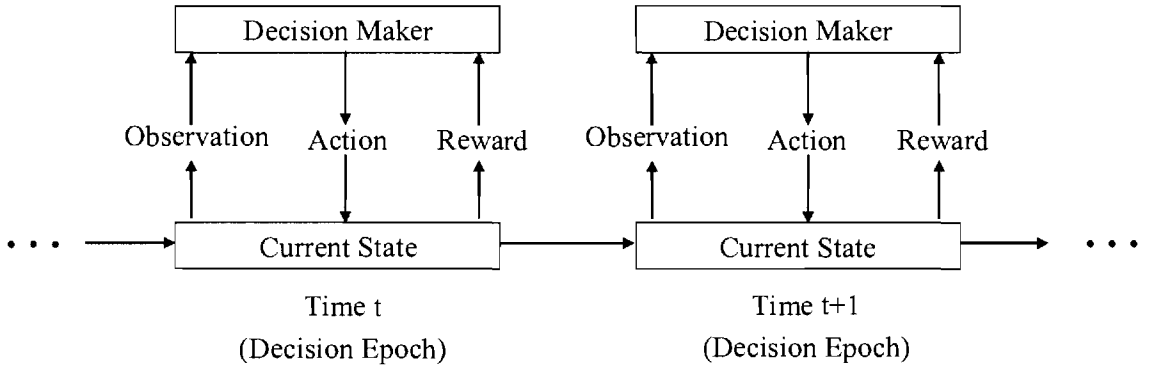


Figure 43: Markov Decision Process.

In a MDP, a decision maker makes decisions at a set of time points, known as decision epochs which can be continuous or discrete. Mathematically, a classical unconstrained, single-agent Markov decision process can be defined as a quadruple (4-tuple) (S, A, P, R) consisting of

- a state space $S = \{i\}$;
- an action space $A = \{a\}$, where the set of possible actions in state i is denoted by A_i, and $A = \bigcup_{i \in S} A_i$;
- a transition probability distribution function $P = [p_{iaj}]$: $S \times A \times S \mapsto P(S)$, where $P(S)$ defines the space of probability distribution over the state space S, and p_{iaj} is the probability of transiting to state $j \in S$ by executing action $a \in A_s \subseteq A$ in state $i \in S$; and
- a reward function $R = [r_{ia}]$: $S \times A \mapsto R$, where r_{ia} defines the immediate reward earned for executing action $a \in A_i \subseteq A$ in state $i \in S$.

A Markov decision process starts from an initial state $i_0 \subseteq S$ and, as an action $a \in A_s$ is taken, transits to the next state j with a probability of p_{i_0aj} defined in the transition probability function P. Then a new action is chosen and executed in current state, resulting in a new transition. In the process, at decision epoch t the state of the system is in i_t depending on the system's trajectory.

As mentioned earlier, the solution to the Markov decision processes is defined as policy. A policy, denoted as π, is a mapping from states to actions, which specifies the action to take for a given state, regardless of prior history. A stationary policy is defined as a policy that does not depend on time but only on the current state. It should be noted that almost all the work related to the Markov decision process is to find an optimal stationary policy. The stationary policy can be further classified into two categories: deterministic policy and randomized policy. A deterministic policy always takes the same action for a specific state while a randomized policy chooses an action a for a state i based on some probability distribution over a set of actions $a \in A_i \subseteq A$. A randomized policy, denoted as $\pi = [\pi_{ia}]$, is a mapping of state-action pair to probability distribution, where π_{ia} defines the probability of choosing action a when the system is in state i.

5.4.2.3 Solution to MDP

A policy is preferred over the other if it obtains a better value of the evaluation criterion which is often defined as some cumulative function of the rewards, such as the expected total rewards, the expected discounted rewards, or the average expected rewards. Assuming the expected discount rewards is employed as the criterion to evaluate the policy, if a Markov decision process starts from state i, the expected discounted sum of future rewards $V(i)$ is given by Equation (3)

$$V(i) = r_{ia} + \gamma \sum_{j=1}^{N} p_{iaj} V(j) \tag{3}$$

where r_{ia} is the immediate reward earned by executing action a in starting state i, γ is the discount factor which has property of $\gamma \in (0,1]$. The goal of the Markov decision processes is to find an optimal policy that maximizes the value function, as shown by Equation (4).

$$V(i) = \max_{a} \left[r_{ia} + \gamma \sum_{j=1}^{N} p_{iaj} V(j) \right] \tag{4}$$

Equation (4) is also known as Bellman optimality equations [Bellman, 1957]. From Equation (4) one can see that the value of a policy depends upon the initial state of the process. It has been proved that for an unconstrained MDP there exists an optimal policy such that for any initial state there is no better option than to follow the policy, i.e., $\forall$ policy π and initial state $i \in S$, $\exists$ optimal policy π^* that $V(\pi^*, i) \geq V(\pi, i)$.

There are three widely used algorithms for determining the optimal policy to a Markov decision process. They are value iteration, policy iteration and linear programming. The value iteration algorithm calculates the value function, given by Equation (4), by finding a sequence of value functions, each one

derived from the previous one. This iteration continues until the value function for the desired horizon is found, or until the value function is converged. Finally, the optimal policy is derived from the maximum value function, given by Equation (5). Policy iteration manipulates the policy directly, rather than finding it indirectly via the optimal value function $V^*(i)$. The first step in policy iteration algorithm is to randomly choose a policy π^0 as the starting point. Then the expected rewards $V^0(i)$ for all states along the Markov process are calculated using π^0. After the value of each state $V^0(i)$ under current policy is known, it may possible improve the value by changing the first action taken. If this is the case, a new policy will be produced based on the value of the state calculated using previous policy. The above steps are repeated until the iteration is converged, and at this point the optimal policy π^* is reached.

$$\pi^* = \arg\max_a \left[r_{ia} + \gamma \sum_{j=1}^{N} p_{iaj} V(j) \right] \tag{5}$$

Dynamic programming, including the value iteration and policy iteration algorithms, have a rigorous process in which the value function needs to be calculated for all the states in each iteration. However, as the number of state variables increases, the size of the state space will typically grows exponentially which is known as the curse of dimensionality. In addition, in complex system operation, multiple agents (subsystems) are often involved when the optimal policy is calculated. This fact further increases the state space and action space of the MDP problem. Moreover, since the interactions exist among the agents, certain constraints may be imposed to the problem and needs to be taken into account. These difficulties cause the dynamic programming formulation to become intractable for solving this type of problems.

Linear programming, with the pioneering work of D'Epenoux [D'Epenoux, 1963], was proposed as one of the approaches to deal with these difficulties. An unconstrained single-agent Markov decision process can be formulated as a linear programming, given by Equation (6):

$$\begin{aligned} &\max \sum_i \sum_a x_{ia} r_{ia} \\ &s.t. \ \sum_a x_{ja} - \sum_i \sum_a p_{iaj} x_{ia} = \alpha_j \\ &\quad x_{ia} \geq 0 \end{aligned} \tag{6}$$

or the first constraint can be written as

$$\sum_i \sum_a (\delta_{ij} - p_{iaj}) x_{ia} = \alpha_j \text{, where } \delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}.$$

The optimization variables $x = \{x_{ia}\}$ corresponding to a policy π are referred as the occupation measure which can be interpreted as expected frequency that action a is chosen in state i. Therefore the occupation measure is essentially a probability measure over the set of state-action pairs (i, a) and it has

the property that the expected total reward to that policy can be expressed as the expectation of the immediate reward with respect to this measure [Altman, 1999], as shown by Equation (6). The policy π can be obtained from the x as:

$$\pi_{ia} = \frac{x_{ia}}{\sum_a x_{ia}} \tag{7}$$

$\alpha = \{\alpha_j\}$ is a measure of initial probability distribution over the states. And thus the first constraint in Equation (6) can be considered as the conservation of probability and is not an external constraints imposed on the problem. This constraint can be interpreted as the expected frequency that state j is visited less the expected frequency that j is transmitted from all state-action pairs should be equal to the expected frequency of starting in state j. The second constraint clearly indicates that the probability of taking action a in state i is nonnegative.

5.4.3 Multi-Agent Resource Allocation

A complex system, such as a naval ship, relies on various subsystems to provide the necessary functions in order to successfully perform the desired mission. To maintain their functionalities, all the subsystems need necessary resources, such as electrical power, chilled water, or fuel, to work properly. These resources are often limited and shared by all the subsystems.

It has been stated before that in order to increase mission effectiveness, a modern ship should be able to reconfigure itself into the state that is most suitable for the situation under consideration. The reconfiguration is accomplished by taking the best action in the current state based on the assessment of the incoming information. Thus, during the ship operation a best course of action needs to be identified and taken in the operation process.

The execution of the actions often consumes resources. Since different subsystems need to work together to realize various functions required to complete the desired actions, they require different amounts of resources to function properly. Therefore, there is a clear need for resource allocation among the subsystems in order to ensure their performance and satisfy the ultimate goal of the system operation. The completion of the resource allocation will reconfigure the ship to a new state most suitable to deal with the situation at hand. Hence, the realization of the best course of action and the resource allocation problem are closely coupled, that is, to find the best course of action a resource allocation problem needs to be solved. It is clear that the resource allocation problem can be formulated as a multi-agent Markov decision process.

Apparently, in the resource allocation problem, the subsystems are coupled regardless their work dependencies because their resource consumptions are constrained by the total available resources. In

addition, the resources may be limited so that not all subsystems can obtain required resource. This implies that coordination must be done among the subsystems when the resources are allocated. Therefore, resource available constraints are imposed to the multi-agent MDP formulation.

5.4.3.1 Constrained Multi-Agent Markov Decision Process

Dolgov and Durfee [Dolgov and Durfee, 2004] formulated a multi-agent MDP with operationalization resource and constraints based on the single-agent, unconstrained MDP given by Equation (6). An agent is said to exhibit operationalization constraints if a particular policy is not operational due to the resource limitation. Based on Dolgov and Durfee's work, ASDL proposed an improved linear programming formulation for constrained multi-agent MDP, shown in Equation (8).

Equation (8) not only can handle the operationalization resource, but also can handle the execution and recyclable resources. The operationalization resources include tools, equipments and personnel, and are reusable and often represented as discrete variables. The execution resources like time, fuel and money are consumable. The recyclable resource such as chilled water of a chilled water system is neither reusable nor consumable, but recyclable.

$$
\begin{aligned}
&\max \sum_m \sum_i \sum_a x_{ia}^m r_{ia}^m \\
&s.t \quad \sum_i \sum_a (\delta_{ij} - p_{iaj}^m) x_{ia}^m = \alpha_j^m \\
&\quad \sum_m \theta(\sum_a c_{ak}^m \sum_i x_{ia}^m) \le \hat{c}_k \\
&\quad \sum_k q_{kl} \theta(\sum_a c_{ak}^m \sum_i x_{ia}^m) \le \hat{q}_l^m \\
&\quad \sum_m \sum_i \sum_a h_{iau}^m x_{ia}^m \le \hat{h}_u \\
&\quad \sum_m \sum_i \sum_a x_{ia}^m (g_{iaw}^m - \lambda_m \hat{g}_w) \le 0 \\
&\quad x_{ia}^m \ge 0
\end{aligned}
\tag{8}
$$

where $\theta(z) = \begin{cases} 0 & z = 0 \\ 1 & z = 1 \end{cases}$, δ_{ij} is Kronecker delta, defined as $\delta_{ij} = \begin{cases} 0 & i = j \\ 1 & i \neq j \end{cases}$ and

p_{iaj}^m represents the probability that agent m transits to state j if action a is executed in state i.

r_{ia}^m defines the reward agent m earns for executing action a in state i.

c_{ak}^m defines the action resource requirements, that is, if agent m requires resource k in order to execute action a then c_{ak}^m =1, otherwise c_{ak}^m =0.

$\hat{c}_k$ defines the total amount of resource k available to be shared by all the agents in the group

q_{kl} defines the amount of cost in type l resulting from consuming a unit of resource k.

$\hat{q}_l^m$ defines the upper bounds on how much cost l the agent m can incur.

α_i^m is the initial probability distribution of the state i for agent m.

$\hat{g}_w$ is defined as upper bound of a recyclable resource w.

g_{iaw}^m represents the amount of this recyclable resource consumed by agent m if action a is executed in state i.

Notice that if there is operationalization resource shared by the agents, Equation (8) can be solved by using the MILP technique. On the other hand, if there is no operationalization resource, the second and third constraints will be taken out. It is clear that the problem will become to a linear program and can be solved using linear programming technique.

5.4.3.2 Resource Allocation Formulation

The resource allocation problem is formulated as a constrained multi-agent Markov decision process which can be solved using Equation (8). The solution to Equation (8) is an optimal policy that specifies the action to be taken by each agent in a specific state. Thus the resources consumed by the agent to execute the action are essentially the solution to the resource allocation problem. Therefore, the resource allocation problem and the policy optimization problem are closely coupled.

The resource allocation process can be detailed described as the step by step procedure below:

Step 1: Identify state and action spaces for each agent

Assume the system consists of M agents which operate independently. The state space S^m ($m = \{1,2,\cdots,M\}$) and action spaces A^m ($m = \{1,2,\cdots,M\}$) of the agents need to be identified. A state of an agent is defined by one or more state variables. In each state of agent m, there is a set of action A_i^m can be taken, and all A_i^m compose the action space A^m.

Step 2: Estimate transition probability function and define immediate rewards

For each agent m, the transition probability matrix $P^m = [p_{iaj}^m]$ ($m = \{1,2,\cdots,M\}$) needs to be estimated. The transition probabilities are often estimated using the historical data or calculated based on

the simulation results. The immediate reward r_{ia}^{m} of the agent for each state-action pair needs to be defined by decision maker based on the expected effect of the action.

Step 3: Identify the resource type, upper bound of each resource and resource required by each agent for each state-action pair

The resources required to carry out the actions should be identified and their types (reusable, consumable or recyclable) need to be recognized. The upper bounds of the resources are required to be known. In addition, the resources required by each agent for each state-action pair need to be identified.

Step 4: Find optimal policy

With the inputs well defined in step 1 to step 3, a constrained multi-agent Markov decision process can be formulated utilizing Equation (8). The optimal policy will be obtained by solving the equation employing the linear programming or mixed integer linear program technique.

Step 5: Resource allocation

After the optimal policy is produced, the resource allocation problem can be fulfilled in the system operation process. At a decision epoch, the optimal policy specifies which action should be taken in the current state, then the resources required to carry out the actions will be distributed to the agent to complete the resource allocation task.

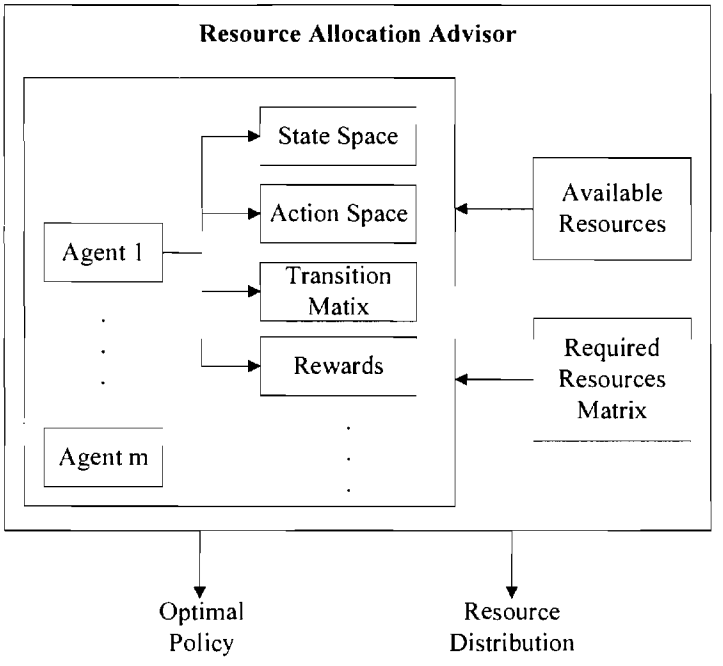


Figure 44: Resource Allocation Advisor.

In the ship operation, in order to increase mission effectiveness and reduce cost, autonomous decisions need to be made. Thus, the decision making associated with resource allocation requires automation. A decision making advisor, shown in Figure 44, is proposed to realize the autonomous resource allocation. This advisor encompasses a constrained multi-agent MDP formulation which can generate the optimal policy used to allocate the resource.

It can be seen from Figure 44 that with all the inputs available, the advisor automates the step 4 and step 5 of the resource allocation process. In the system operation, some event may happen, such as damage occurrence or mission change. In this case, the associated inputs of the resource allocation advisor should be updated, and then the new optimal policy is calculated to direct the resource allocation process. This is illustrated in Figure 45.

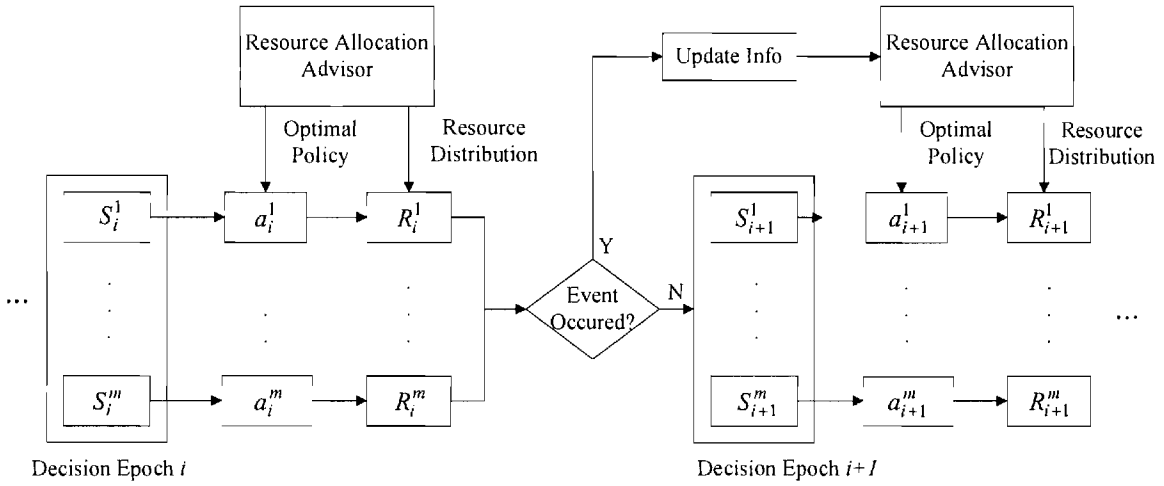


Figure 45: Resource Allocation When Event Occurs.

5.4.4 CASCADE

The question to be addressed in this section is how to design a controller that takes into account the problem of cascading failures. As presented previously, a solution to the problem of cascading failures in complex, highly interdependent and dynamic systems is necessary to ensure graceful degradation. This section describes an attempt at creating a methodology for including concerns of cascading failures in the reconfiguration of a highly interdependent and dynamic system.

The key enabler in this formulation is the use of faster-than-real-time computing, using surrogate models of the different systems to perform probabilistic studies and forecast the likelihood of failure of each configuration under uncertain conditions for a given scenario. Two approaches are studied. The first one makes use of a Bayesian-Network of the operational conditions to provide the calculation capability over which a Genetic Algorithm (GA) selects the best family of solutions that are then presented to the operator. The second approach uses a simplified interrelationship dynamic graph representation of the

network of systems to estimate the likelihoods of inducing failure. This is a continuous time simulation using probabilistic estimates of the tendency that each component has on inducing a failure on another. A GA is then used to select the most robust configuration.

The surrogate models, or metamodels, are created using Design of Experiments (DoE) techniques, by running a 'smart' number of cases that provide maximum knowledge for a minimum amount of executions. The form of the surrogate model is a Neural Network due to the non-linear behavior of the space under consideration. The Bayesian-Network (BN), also known as Belief-Network, is used, along with the probabilistic results from the surrogate model, to perform a quick evaluation of all the possible reconfiguration states in which the system can be aligned. In the second approach, the likelihoods of inducing failure for the dynamic interrelationship graph are obtained from more complex physics-based models of the system.

The Genetic Algorithm is programmed to be a Pareto Optimality searching algorithm [Coello Coello, 2001] that attempts to capture the set of non-dominated solutions. There are many options to the type of algorithm that can be used; the solution proposed here will be a variation on the Niched-Pareto Genetic Algorithm [Horn, 1991]. A set of non-dominated solutions means that all the solutions are equally good, and an improvement in any metric cannot be obtained without diminishing the goodness of the solution in another metric; the selection of the optimal solution is dependent on the weight that the operator gives to each of the metrics, effectively condensing the Pareto frontier to a point. In this case a metric is anything that exemplifies the goodness of that distribution of resources, whether that be efficiency, e.g., minimum fuel burn, or survivability, e.g., maximum likelihood of avoiding cascading failures.

5.4.4.1 Bayesian Networks

Figure 46 depicts how Microsoft's MSBNx [Kadie, 2001] was used to create the demonstration Bayesian-Networks for a simplified model of a representative number of operational decisions that a crew from a next-generation surface combatant would have to select from. The purpose of this investigation was to evaluate the capability of overlaying a Genetic Algorithm over a Bayesian-Network to obtain a series of operational recommendations that can then be provided to the user for selection.

The results obtained were satisfactory in the sense that it is clear that such an approach is feasible, the problem lies in the fact that the Bayesian-Networks do not allow for feedback between the nodes, incapacitating the proper representation of the system at hand. In order to overcome this obstacle, alternative representations of the system were attempted, but without avail. For this reason, the decision was made to represent the operational options and interdependencies of the IEP through an alternative approach.

Bayesian-Networks still can serve a purpose in providing a framework on which to base higher-level types of decisions, as for example, which end-service loads to operate, e.g., radar vs. motors (awareness vs. mobility) and other tradeoffs of the like, but not reconfiguration decisions at the component level such as the valves and switches because their interdependencies need to be accurately modeled.

There remains the possibility of merging these two approaches at a later stage. The dynamic probability analysis would feed information to the Bayesian-Network to account for the higher-level reasoning and the interdependencies and probabilistic failure analysis would be performed by the MATLAB™ code.

Figure 46: Partial screenshot of a Bayesian Network from MSBNx. [Coello Coello, 2001]

5.4.4.2 Probabilistic Simulation

The MATLAB™ program described in this section was developed to provide a framework on which to stochastically evaluate the likelihoods of cascading failures through a heterogeneous, interdependent system [Dueñas Osorio], [Cruicitti, 2004], [Watts, 2000]. The code was christened the Computational Analysis of Stochastic Cascading Actions for Design and Evaluation (CASCADE).

CASCADE runs a Monte Carlo Simulation of different cascading events that can propagate through the system. In order to do this, CASCADE requires two pieces of information, the Component Dependency Array (CDA) and the Dynamic Dependency Probability Function (DDPF).

The CDA specifies all of the system's components and their associated dependencies on the other system components; e.g., if the Ship Service Converter Module (SSCM) depends on the Power Supply 1 (PS1) and the Heat Exchanger 2 (Hx2), the list for the SSCM will contain the index for PS1 and Hx2. This dependency array can be obtained by analyzing which components provide a service or resource to which other components. The DDPF is the probability of inducing failure as a function of time when the components in the CDA fail. The DDPF can be obtained by running a Monte Carlo Simulation (MCS) of a series of higher fidelity codes that model the physics of the dependency of the component in question with the ones in the CDA, or by gathering experimental data from hardware. There will be a DDPF for each entry in the CDA. The DDPF also contains the probability that it will induce a failure, effectively the area below the probability density function (PDF). In this case, the PDFs have been assumed to be Weibull functions because they can be fully described with just two parameters, only take positive values and have been historically observed to accurately represent failure rates. It is important to notice that the approach allows for any type of PDF and even for the use of different types of PDFs for each component at the same time.

The assumptions behind the approach are the following:

- Failures propagate sequentially, i.e., if component A is connected to B and B is connected to C such that C depends on B and B depends on A, a failure in A will not induce failure of C without failing B first.
- Probability of failure is independent of the order of failure for components on which there is a redundant dependency, i.e., if component A depends on both B and C to fail, for A to fail, whether B fails first and C second, or vice versa, is unimportant to the distribution of failures for A or at least the DDPF was created with a representative set of failures for B and C.
- The distribution of modes of failure used to obtain the DDPF is representative of the failures that will be exhibited by the network, i.e., assume A depends on B and C, if during the simulation to obtain the DDPF for component A, components B and C are failed in a manner that is representative of how they will be failed by the components on which they depend.

A proof-of-concept model containing both power distribution and thermal control components was developed to demonstrate the approach. The components and their dependencies are represented in an n^2-diagram shown in Figure 47.

The dependencies in this model are electrical, fluid and thermal transfer dependencies, and the selection of the representation of the system was such as to separate the two networks as much as possible

and ease in the definition of the interdependencies. It would be possible to merge the Heat Exchangers with the components they cool, but that would mean that components have much more complex DDPFs. The more complete the decomposition, the easier it is to represent each dependency, but with the understanding that more interdependencies must be specified. Figure 48 is a sample result from CASCADE. The x-axes are time in units of seconds. The left column represents the inverse cumulative distribution function (CDF), and the right column the probability density function (PDF). These results were obtained by running 10,000 cases. The inverse CDFs can be used to estimate for a given certainty how long the system will be operational. Say, there is a 90% certainty that the pump will be running up to 3 seconds after the initial event occurred.

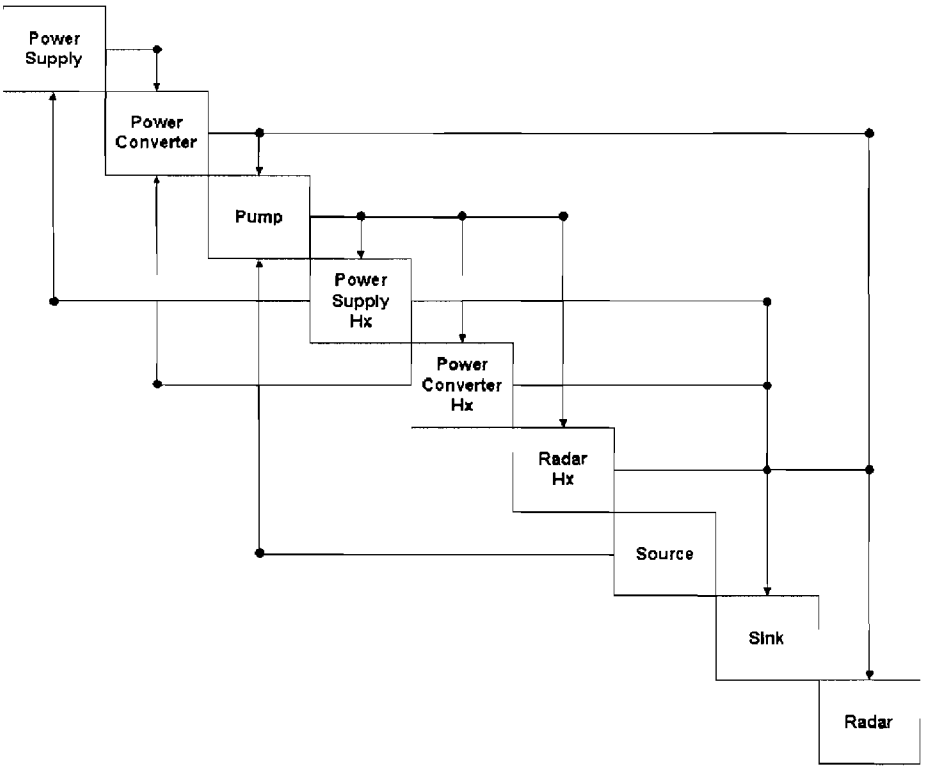


Figure 47: Interdependency system model in n^2-diagram form.

The Genetic Algorithm will switch components on and off, open and close valves and electrical switches, effectively changing the CDAs and DDPFs, in an attempt to maximize the operational time of the components with the highest priority. The family of solutions that offer the best configurations will then be presented to the operator for selection. This approach only investigates the survivability or operability of the system. Other considerations would be the efficiency, and the actual feasibility of that given configuration. For this reason, it is important to account for the distribution of the limited number of resources and which devices can actually be operated, and at what settings. In order to account for this, a Resource Allocation Algorithm has been developed.

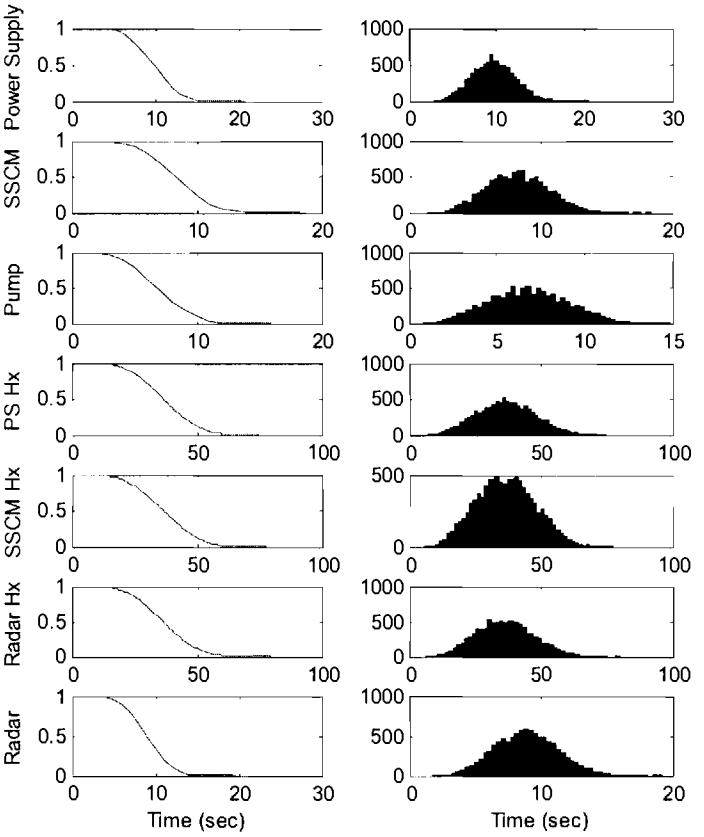


Figure 48: Sample results from CASCADE.

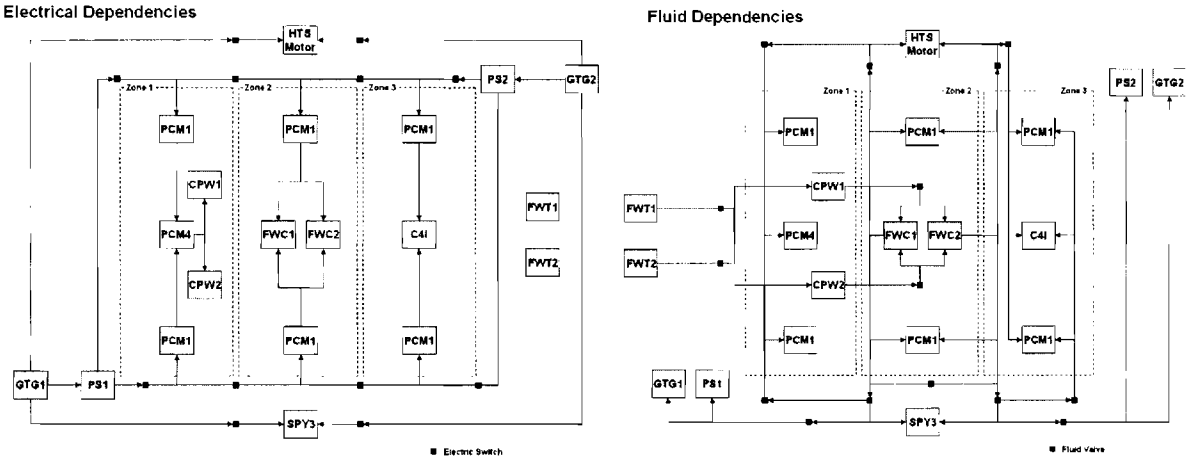


Figure 49: Model used to study the performance of CASCADE.

A more complex model was later developed and implemented to test the scalability of the method. Figure 49 depicts the new proof-of-concept model; it includes a more realistic architecture, with 20 components and a reconfigurable network with switches and valves. The method scalability approximates an NP-hard problem, indicating that the method needs to be adapted to scale in a more linear manner

because the number of components of a realistic application will increase by approximately 2 orders of magnitude.

5.4.5 Human Machine Interface

As stated earlier the initiative is an effort to develop methods to address the difficulties that arise in complex interdependent reconfigurable systems, including the complication introduced by human supervision of the automation system. The complexity is a product of the interdependencies of the subsystems and the reconfigurability of the overall system. Each subsystem has a set of requirements for each mode of operation it can run in. These requirements are met using available ships resources. Many subsystems both provide ship resources and consume other resources. Redundancy in subsystems is desired for all critical functions of the overall system, including redundancy in resource paths throughout the overall system. Because the overall system is relatively large in scope much of the analysis is performed on functional decomposed parts and not on the overall system as a whole. This practice alone can not capture the emergent behaviors at the overall systems level. To fully understand the overall system these behaviors must be studied. An analysis can be developed using each of the subsystem analysis to provide a picture of the overall system in a given state in time. Since the behaviors propagate through the overall system in time a time domain analysis is appropriate.

A simulation is essentially an analysis performed over time. The ability to accurately simulate a system correspondence to the ability to predict the response of the actual system given an identical initial state and stimulation. This capability can be of great benefit to the engineers trying to understand the system while in the early design phases. For these reasons the efforts to understand the interactions of the physical system are being addressed using simulation or more accurately in this case an integration of subsystem simulations. However, the use of a simulation for this system brings rise to more questions and challenges. First, how to integrate all of the subsystem simulations and in some cases which subsystems simulations should be included for which studies. Second, how to represent the human interactions in the overall simulation. These questions among others make this a nontrivial task.

Human-in-the-loop simulation is a simulation scheme involving humans for either decision making or supervision. The question proposed earlier, how should the human interactions in the overall simulation be represented is a crucial question to answer. The human interactions could be accounted for in a number ways, nonetheless there should be a reason for choosing one option over another. In this case the simulation is intended to provide more information earlier in the design process when decision must be made. Eventually these systems will be operational with active human interactions stimulating the systems and thus bringing to life emergent behaviors. One method for accounting for this contribution is to either develop or utilize an existing human behavior model. Human behavior models exist and they

have been used successfully for modeling group behaviors. For example human behavior models have been used to help understand and design for the behavior of a group trying to escape a burning building. The benefit from this approach is a self contained simulation. One could start the simulation and allow it to run over night without user interaction. This may seem appealing especially from a cost point of view, however, an alternative could simply be to include actual human interactions. The latter option does not completely dismiss the previous, and it has the benefits of potentially obtaining a more true interaction being a closer resemblance of the actual physical system. It the case that a human behavior model became of higher value any mechanism used to capture the human interactions from the option could still be utilized to collect data for a human behavior model. This type of simulation, a simulation utilizing actual human interactions is called human-in-the-loop simulation. This type of simulation brings with it both a number of new advantages and a number of new restrictions.

Before choosing human-in-the-loop simulation will make sense, its advantages and disadvantages need to be understood. Starting with the disadvantages, human-in-the-loop simulations can potentially place some very difficult restriction on the simulation computationally speaking. If humans are to interact with the simulation in a realistic manner the simulation must be capable of running in real-time or better. This restriction is a constraint on the fidelity capabilities of the simulation with respect to the hardware capabilities. Depending on the system being modeled the simulation could expect varying degrees of interaction and thus requiring various degrees of man power to run the simulation. This argument alone may drive the solution to a more of a hybrid system. A hybrid could allow the engineers (users) to adjust the man power requirements a simulation may need during run time. Certainly there could be more disadvantages, nevertheless in many cases a disadvantage could be converted with the right mix of technologies. Which technologies will be discussed later.

With the introduction of disadvantages several advantages are also introduced with human-in-the-loop simulation. How to design, build, validate, and fully utilize simulation code is still a research topic at the university level? The true bottom line product of a simulation for engineering purposes is information. It then seem appropriate to ask what information is of interest. In this case the simulation is part of a design methodology, thus design decision will be based on all the information available at decision making time. Which time is consequently rarely a time when enough information is known. The real question that the simulation is hoped to help answer is how will each decision affect the finished overall system. For a system intended to be operated by humans it follows that a simulation of a system, should also be design to be operated by humans in a like manner. Thereby, providing the ability to predict the affect of early design decision to the overall system including the human contributions in the simulation environment.

Design by nature tends to be an iterative process. Yet the conventional design processes tends to first decompose the system and iterate subsystems independently from the overall system. Human-in-the-loop simulation in general can be an enabler around this practice, and provide more opportunity to explore more options while still in the early design phases.

An interface (human machine interface or HMI) is required to enable interactions between the simulation environment and the human participants. Generally requirements for this interface are derived from certain expectations of human users. These expectations are a mixture between systems the users have operated in the past and a desire for convenience and efficiency. The following are some points to consider:

- The simulation must run in real time
- The simulation must continue running even if the operator declines to interact or is unavailable
- The human machine interface (HMI) must be accessible and responsive
- The HMI must integrate seamlessly with the simulation environment
- The HMI must have a minimal contribution to the computational burden of the simulation
- The HMI must be general enough to handle the full range of interactions the simulation requires
- The HMI must be flexible enough to evolve with the design process
- The HMI must be an efficient interface suitable for an end user without expertise
- The HMI must be able to portray a maximum amount of information without over whelming the user
- The process of extending or modifying the HMI should not be overly complicated or esoteric.
- Most simulation software packages were not intended for human interaction during run time, thus a method must be developed to extent this functionality.

The HMI needed to be lean, simple, and yet very powerful, and feature rich. The HMI needed visualization capabilities for efficient communication of potentially large amounts of information. It needed a common protocol or standard means to allow communication with popular simulation software packages across multiple engineering domains. It was desirable to run the HMI on a separate machine from the simulation environment, thus requiring a network capability. In response to the desired

capabilities and requirements a web-based application framework was proposed and developed. Utilizing modern web-based technologies the HMI proofed it's self and continues to show great promise.

The study of the behavior of human-in-the-loop control is essential because the Navy's culture does not permit automation systems to work unsupervised, and therefore it is crucial to see how the dynamics of that interaction will play.

5.5 Crew Modeling

The goal of this model is to provide a representative crew model that will allow the US Navy to perform automation/manning tradeoffs in a time-domain simulation environment. The goal of optimizing manning for future naval combatants leads to the need of analyzing the tasks that can be performed by the crew and how it can be optimized to maximize mission effectiveness while abiding to the quota on the number of sailors. In order to do this, it is essential to create a model that can simulate the capabilities of sailors and their interactions with the ship and themselves. As the ship becomes more automated the task burden on the sailors diminishes and the crew size can be reduced. The question is where is the point of diminishing returns or when does it become more effective to keep sailors in the ship both from a mission and a survivability stand point? The tradeoff between platform survivability and crew casualties needs to be addressed. In summary, there is a need for a method to estimate the optimum, most survivable crew, in terms of both quality and quantity and the impact that technologies have on optimal manning. The following methodology was a study to attempt to formulate a methodology for addressing these concerns.

5.5.1 Simulation Process

The underlying approach is to treat the crew as a resource, very much like electricity and cooling was abstracted in the previous formulations. The critical manning condition for ship design are damage control and unexpected maintenance at sea. Damage control is the most critical because it is when the ship's performance is degraded the most and the crew has the highest burden in terms of labor. This is also the hardest condition to simulate because it requires a detailed time-domain simulation of the behavior of the crew, unlike task decomposition approaches that are satisfactory for non-time domain studies. For this reason, the team studied the possibility of creating a method to develop crew simulation environments in situations of damage control. The goal of this method was not to create highly detailed models, but models that were representative of the dynamics exhibited by crew and the ship for the purpose of capturing the manning requirements of different scenarios and tradeoff technology for manning. The general simulation process is presented in Figure 50, this is a time domain simulation loop that is constantly repeated at a fixed time step. But before the simulation can take place, the modeling environment has to be set up.

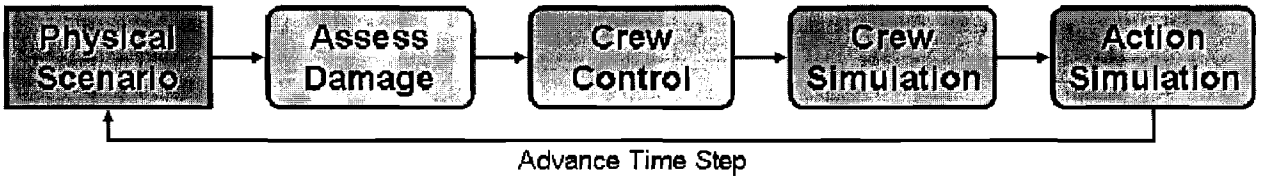


Figure 50: The crew modeling process

5.5.2 Modeling Approach

The first step in the modeling process is to translate the characteristic information to matrix form. Four matrices are used to characterize the crew model. These are the Crew Information Matrix (CIM), the Crew Communication Matrix (CCM), the Ship Information Matrix (SIM) and the Ship Connection Matrix (SCM). The CIM contains the information on crew status, location and abilities. The CCM contains the information on who can communicate with whom. The SIM contains the status information on compartments and components. The SCM contains the information on which compartments are adjoined. Figure 51 contains an example of the four matrices and the type of information that they contain.

$$Status_{Comp}(t_i) = 1 - (1 - Status_{Comp}(t_{i-1}))\exp\left(-\frac{Status_{Crew}(t_{i-1}) \cdot Ability_{Crew}}{ComplexityFactor_{Comp}} \cdot (t_i - t_{i-1})\right) \quad (9)$$

$$Status_{Fire}(t_i) = Status_{Fire}(t_{i-1})\exp(-Status_{Crew}(t_{i-1}) \cdot Ability_{Crew} \cdot FFCC_{Comp} \cdot (t_i - t_{i-1})) \quad (10)$$

$$Status_{Flood}(t_i) = Status_{Flood}(t_{i-1})\exp(-Status_{Crew}(t_{i-1}) \cdot Ability_{Crew} \cdot FCCC_{Comp} \cdot (t_i - t_{i-1})) \quad (11)$$

The actions the crew can perform have been abstracted using simple exponentials that have been abstracted and simplified to ease in developing the models and the computation requirements. Three tasks, fixing a component, firefighting and de-flooding, have been modeled using the following equations. The compartments' Fire-Fighting Capability Coefficient (FFCC) and Flooding Control Capability Coefficient (FCCC) are abstractions of the ease with which the crew can perform firefighting and de-flooding tasks in that compartment, higher numbers indicate that it is easier for the crew to achieve its goals.

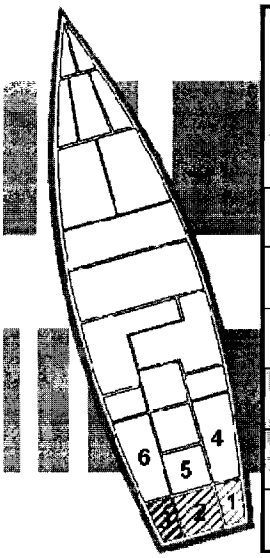
Crew Information Matrix (CIM)

ID (Crew)	Location (Comp ID)	Health Status	Abilities			
			Mechanic	Fire	...	Flooding
1	2	1.00	0.10	1.00	.	0.10
2	2	1.00	0.40	0.40		0.40
3	5	1.00	0.40	0.10	.	1.00
4	4	1.00	1.00	0.10	.	1.00
5	4	0.50	1.00	0.10	.	1.00
.		...	.	...	.	.
N_{Crew}	N_{Comp}	0 < X < 1	0 < X < 1	0 < X < 1		0 < X < 1

Crew Communication Matrix (CCM)

ID	1	2	3	4	5	.	N
1	1	1	1	1	0	..	1
2	1	1	1	1	1	...	1
3	0	0	1	1	0		0
4	1	1	1	1	1	..	1
5	1	1	1	1	1	...	1
	.	.	..	...	..	...	..
N	1	1	0	1	1	..	1

Ship Information Matrix (SIM)



ID	Status		Resources	
	Fire	Flooded	FFCC	FCCC
1	0.80	0.00	0.50	0.50
2	1.00	0.00	0.70	0.70
3	0.40	0.80	0.50	0.50
4	0.00	0.00	0.40	0.40
5	0.00	0.00	0.60	0.60
...		.		..
N_{Comp}	0 < X < 1	0 < X < 1	0 < X < 1	0 < X < 1

Ship Connection Matrix (SCM)

ID	1	2	3	4	5	...	N
1	1	1	0	1	0	.	0
2	1	1	1	0	1	.	0
3	0	1	1	0	0	.	0
4	1	0	0	1	1	.	0
5	0	1	0	1	1	.	0
	.	.				.	..
N	0	0	0	0	0	.	1

Figure 51: Modeling the crew and ship using matrices.

5.5.3 Proof-of-Concept Model

A ten compartment, three components and six sailors proof-of-concept model was developed in MATLAB™ to test the model. The only action modeled was machinery repair. The communications was not implemented at this preliminary stage. The results of the simulation were promising and demonstrated the feasibility of modeling the crew-ship interactions. The modeling process was presented in a preliminary progress report, but it was requested that further efforts not be invested in it because other researchers were pursuing similar goals and it was agreed that the integrated framework could include these more advanced models at a later stage. Figure 52 is a depiction of the model used as a proof-of-concept. Figure 53 is a time history of the machinery status as it is being repaired by the sailors. Sailor 6 that was in room 4 repairing turbine 2 left after it was operational at 95% status. The intelligence assigned

to the sailors was rudimentary and could benefit from cognitive models that would make their behavior approximate that of real human beings. It is important to keep in mind though that the goal of this project is not to model a human being's decision making process exactly, rather include the capability of integrating models that do to obtain a holistic view of the system's behavior. A crew model could not be obtained in time to integrate, but preliminary tests with the model presented in this section demonstrate that it is possible to include such simulations in the integrated framework.

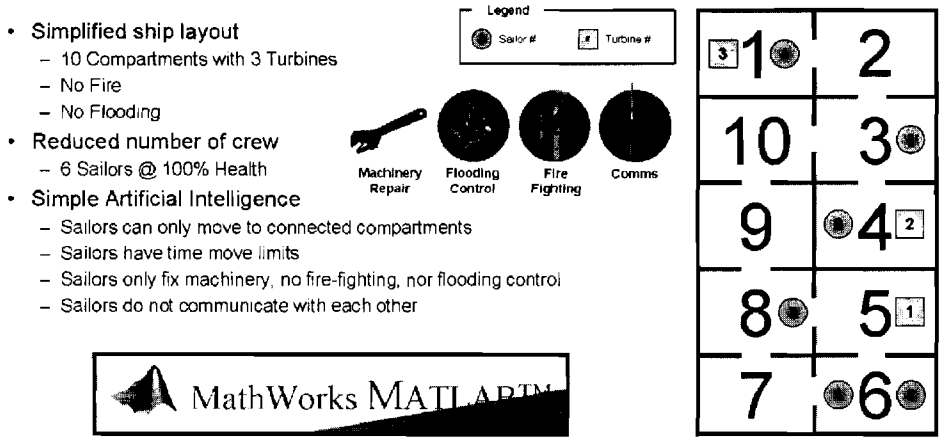


Figure 52: Crew model proof-of-concept formulation

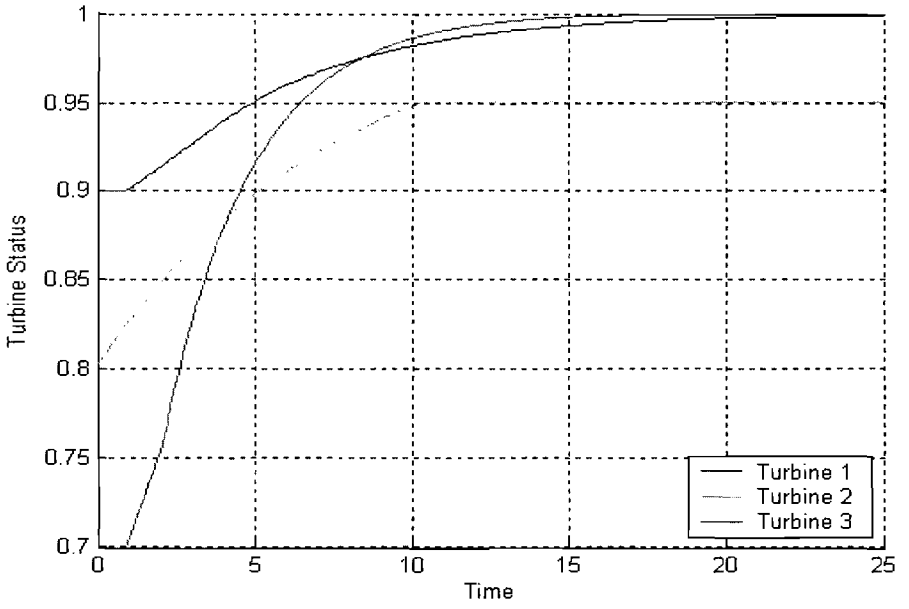


Figure 53: Results from proof-of-concept model

5.6 Accelerating the Analysis

Hundreds, if not thousands, of configurations must be studied under many different conditions to ensure that the system will be robust. This forces the designer to execute extremely longs run times of the simulation environment. Therefore there is a strong need to accelerate the capability to perform analysis of these alternatives. Two general options as available, one is to use hardware to perform faster computations, the other is to create simpler models that accurately describe the behavior of the simulation environment.

5.6.1 Field-Programming Gate Array Boards

Field-Programming Gate Arrays (FPGA) are composed of programmable logic components that can duplicate logical as well as simple mathematical functions. The result is an optimized piece of hardware that can be aligned to simulate a computational process. FPGAs have been used to simulate dynamical processes and reduce their run time by approximately two orders of magnitude [Bastos et al., 2005]. The detriment is that programming the arrays is not a simple task and translators for doing so are not available for all modeling software. The FPGA approach is currently being evaluated as a possible solution to the problem of analysis acceleration, particularly after discovering research conducted by Professor Monti from the University of South Carolina [Bastos et al., 2005], but no experimentation has been conducted at this time.

5.6.2 Surrogate Modeling

A surrogate model – also called a meta-model in some disciplines – is an approximated model of its original model. Since the surrogate model is an approximated model, its accuracy is defined by the tolerance used when it is generated. Accuracy is not guaranteed for extrapolations outside the range of inputs used to generate the surrogate model. Therefore, the surrogate model needs a priori knowledge of what inputs and outputs are needed to be investigated, the ranges of those inputs and outputs, and for what purpose the surrogate model will be used. Then, why is the surrogate model used? Firstly, it can run considerably faster than the original model, and it is suitable to generate physics-based models which are mostly made of custom legacy codes or commercial simulation tools. Secondly, the model can be regenerated in the form of a relatively simple mathematical expression when the model is mathematically complicated or only possible to be expressed by an executable file of a simulation tool. The surrogate model can be a great model replacement when there is a black box system or model and there only is a set of input-output samples as the result of experimentation.

What are the advantages by applying surrogate modeling to the integrated simulation for a dynamic complex system? As mentioned, the simulation speed will be improved, which is especially an important

characteristic; the original model is encapsulated. The surrogate model works like the original model, but the mathematical expression in the surrogate model completely conceals the information of the original system. It means that a system model can be distributed everywhere in the form of a surrogate model even if it is classified information; surrogate models are less dependent on the simulation environment. When an original model is distributed, its simulation environment, typically a commercial simulation tool, should be distributed and installed so an additional license may be needed to be purchased if the computer doesn't have one. On the other hand, a surrogate model can be distributed by the short code of ANSI C/C++ or Visual Basic language, Excel spread sheet, or even a compiled exe file or a COM object which is almost independent of platforms, frameworks, or simulation tools. The surrogate modeling will broaden the model's compatibility and portability.

Then what are the costs of the surrogate modeling? A surrogate model is generated by training sample data made of input-output set extracted from the original model. As more accuracy is needed, the size of the data required by the surrogate model increases, which leads to more intensive computational requirements. Especially for a complex model, the sample data size grows exponentially as the number of input variables increases so it becomes practically impossible to apply the surrogate modeling to a model with dozens of input and output factors. When it is applied to a dynamic model, the generation of sample data often becomes a more demanding task than that of a static model, since there are not many dynamic simulation tools that provide direct access and control of all the input and output factors needed to generate the samples. Then, some indirect approach may be needed in order to extract those factors, requiring more computational effort. This section will describe what type of surrogate model will be chosen, and how it will be applied to complex dynamic models.

5.6.2.1 Artificial Neural Networks

There are various types of surrogate models such as second order polynomials or other multi-order polynomials, Kriging models, or various types of artificial neural networks (ANN). Among those, ANN can be the best choice as a safe starting point due to its great adaptability and capability of approximating nonlinear functions. ANN is a mathematically well-proven universal function approximator which can theoretically approximate any linear or nonlinear function once a dense enough data subset from an original function is provided for training [Mandic and Chambers, 2001]. For approximation of dynamic system or function, ANN especially needs the feedback of output values at the previous time since the future output of a dynamic system are determined in nature by the current and previous information and status of the system. These neural networks specified for dynamic systems are called recurrent neural networks (RNN). Figure 54 describes the structure of a RNN with one dimensional output.

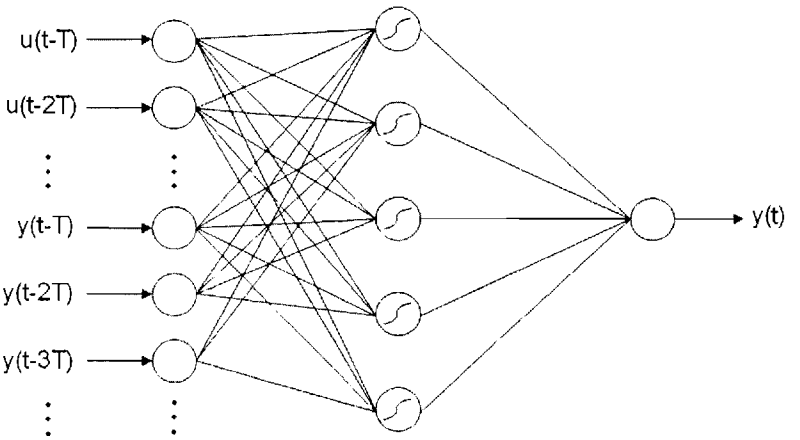


Figure 54: Recurrent artificial neural network

After its training is done, the RNN can approximate the output value y(t) with respect to the time advancing with the time step T. However, there is one shortcoming with RNN, which is that only one fixed time step should be used once the RNN is trained with a certain time step. Therefore, RNN is not applicable to a simulation environment in which a user can adjust the simulation time step when it starts or wants to use a time step varying algorithm such as Runge-Kutta method. With this in mind, one possible alternative to the RNN is the ANN with the output derivatives as output values [Wang and Lin, 1998]. Considering that the y(t) in RNN is the system state variable, x(t) in most dynamic systems, the output derivatives are the state derivatives in the application. Figure 55 is the graphical description of the ANN with state derivative output.

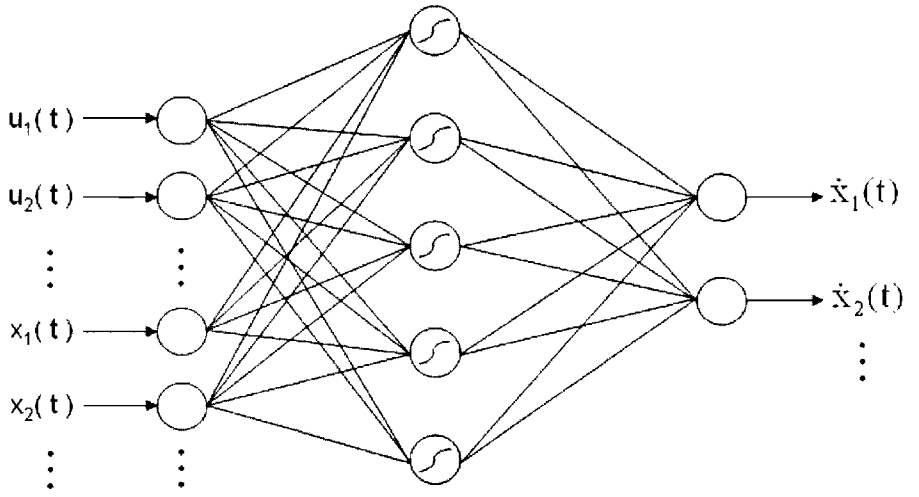


Figure 55: ANN with state derivatives as output

One more difference with RNN is that a numerical integration should be performed to obtain the state variable for the next time step; on the other hand, RNN doesn't need the numerical integration to obtain the state value of the next time step. The ANN with state derivative output for surrogate modeling has been chosen, since the advantage of varying time step may be an important function in the integrated simulation environment.

5.6.2.2 Complex system model decomposition

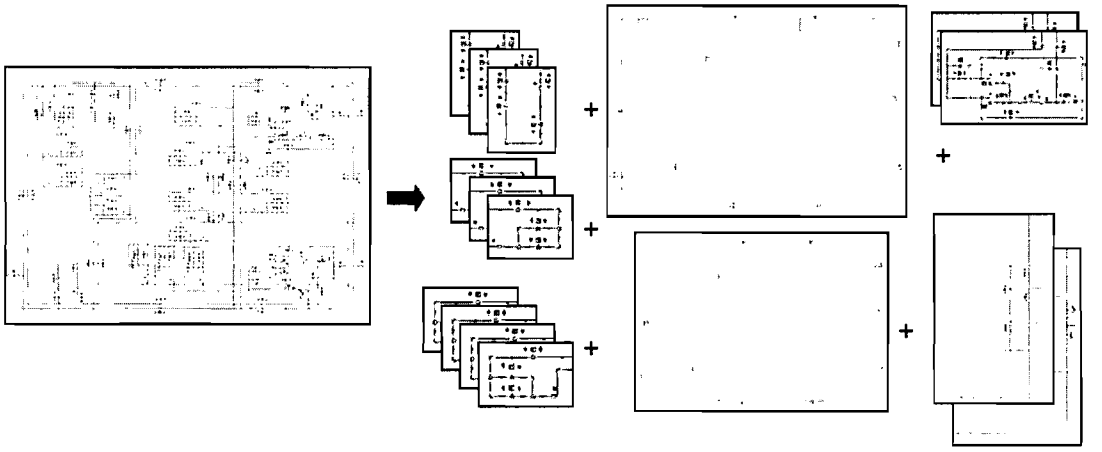


Figure 56: CW-RSAD example of model decomposition

There are about 98 input factors (20 state variables and 78 input variables) in the CW-RSAD without the thermal analysis. The electric power system is another big dynamic model in the whole ship system; it should have, of course, a similar order of the number of input factors. If each factor in the CW-RSAD has five levels to make the sample data, the number of samples will be $598 = 3.15544 \times 1068$ in a full factorial combination, which is a surreal number in engineering sense. There must be some physical constraints like mass conservation or the fact that pressure always decreases to downstream, this condition will not do much in decreasing the astronomical size of the sample data. However, what if there are smaller models with six to seven input factors, and they are connected to form the original model? If the original model is decomposed into twenty smaller models with 6 input factors, the number of samples is $20 \times 56 = 312{,}500$, which is a quite moderate number of runs for today's computing power. If the same physical constraints are considered, the number of samples will be less than that. A new approach for making a surrogate model of a complex system model is based on the above idea. Firstly, the complex system model is decomposed to several smaller and manageable models. Then, a surrogate model is built for each smaller system models. Finally, the smaller models are reconnected and glued based on continuity rules such as mass conservation, Kirchhoff's law, and so on. Therefore, the surrogate model of

the entire system model will be made of modular ANNs and a set of continuity equations to be solved for each time step. Figure 56 and Figure 57 are examples of a possible decomposition for CW-RSAD and the algorithm for reconnection applicable to one of the closed-loop piping network and 16 service load networks in CW-RSAD.

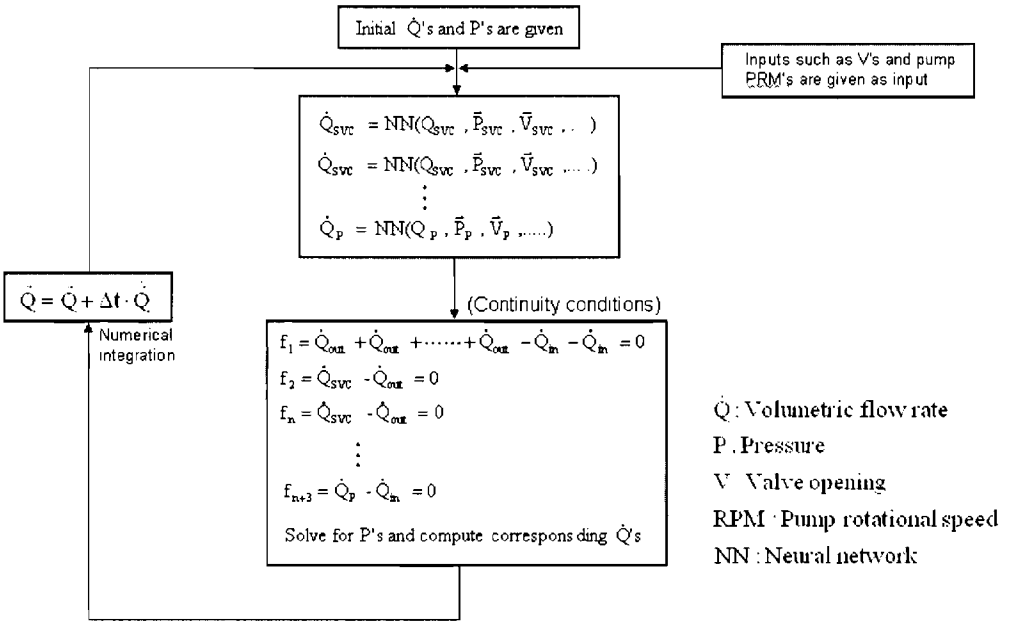


Figure 57: CW-RSAD example of the complex-system surrogate modeling and simulation

Again, a computation burden arises from the modular surrogate modeling of a complex system model. For each time step, the set of continuity equations should be solved numerically, and this computation can give a significant effect to the computing speed. So it is very important to find a very efficient equation solver for the continuity equations.

6 Implementation

Once the theory and methodology has been developed, the implementation of these ideas must be conducted to test their validity and verify that they do indeed produce the expected results. This section is concerned with describing the implementation of the engineering processes that the IRIS team developed.

6.1 Systems Engineering

An IRIS design methodology must combine several methodologies developed at ASDL for the design of complex systems that have been adapted to fit the needs of naval engineering. These being with problem understanding, formulation and definition, using standard system engineering methods as well as approaches and tools developed at ASDL.

6.1.1 SWARMing

The first step was a SWARMing (System Wide Assessment and Research Method) exercise that encompassed a literature search and problem definition where the IRIS team familiarized themselves with the specific ship systems through system decomposition, identified system interrelationships, and determined current state-of-the-art naval engineering tools, methods and techniques.

This task was particularly crucial since the students researching the problem were not familiar with ship design, having a stronger background in aerospace and systems engineering. The result of this process was an extensive literature review and a better understanding of the tradeoffs to be performed.

6.1.2 Knowledge Management

The information gathered through SWARMing was then cataloged into a database using a tool called DSpace, an open-source, web-based digital repository that allowed the IRIS team to set access control to the documents while including a quick search and retrieval function. This proved invaluable throughout the life of the project due to the breath and depth of the information required to accomplish the tasks that will be described later. In an effort to fully understand such a complex system, an interrelationship digraph showing critical subsystems links was created in addition to a representative system decomposition.

6.1.3 Quality Function Deployment

Using the information gathered along with several process tools, the main design drivers can be identified early in the design process through Quality Function Deployment (QFD) and input from experts in the field. A survey to gather experts' opinions was developed using IDEACore's web-survey software WEBMine. The survey can be distributed to naval engineers and Navy personnel operating the state-of-the-art platforms. This allows the designers to obtain the assessment of different requirements and technologies from the people in the field. The customer requirements and engineering characteristics can then be populated in the QFD and ranked according to the experts' opinion. At this stage the survey has not been released, but tests have been conducted to demonstrate its feasibility. The next step was to study the breath of possible options that could be introduced to the IEP design.

6.1.4 Interactive Reconfigurable Matrix of Alternatives

A preliminary matrix of possible subsystems was populated using information gathered from the literature leading to an impossible number of combinations to be evaluated for the IEP. In order to facilitate the intelligent reduction in design combinations, an interactive morphological matrix tool was created. The Interactive Reconfigurable Matrix of Alternatives (IRMA) allows the designer to quickly compare

different design alternatives. It is easily expandable to include as many design alternatives as needed while providing an intuitive visual interface. With the main design drivers identified and a virtual system simulation created, the designer can identify the optimum ship solution. However, without an accurate virtual ship system model, the process fails. The development of the comprehensive system model is the present focus of the IRIS design team.

6.2 Reduced-Scale Advanced Demonstrator Model

Once the qualitative process has been finalized, and a reduced set of promising solutions have been identified, quantitative methods need to be implemented to capture more accurately the competing nonlinear effects that complicate further qualitative analyses. For this purpose, the development of an integrated M&S environment that clearly captures the system dynamics and interdependencies needs to be created, validated and verified. The following section describes the development of such an integrated environment for the purpose of method development, validation and verification. The baseline for the model was the Chilled Water Reduced Scale Advanced Demonstrator (CW-RSAD), over which the remainder of the architecture was developed.

The CW-RSAD is a reduced-scale model of two zones of the Arleigh Burke chilled water system (Figure 58) and is located at the Naval Surface Warfare Center in Philadelphia. The RSAD was originally constructed to investigate the component level intelligent distribution control system which is employed to achieve reliable unmanned control of shipboard auxiliary systems. It consist of 4 pumps, 2 chiller plants, and 16 service loads which are the units of equipment cooled by the chilled water system [Scheidt, 2002]. It also contains 2 expansion tanks with the capacity to deliver 40 gpm of chiller water. The RSAD utilize a vertically offset main loop to distribute chilled water to the 16 service loads [Fairmount Automation, 2006].

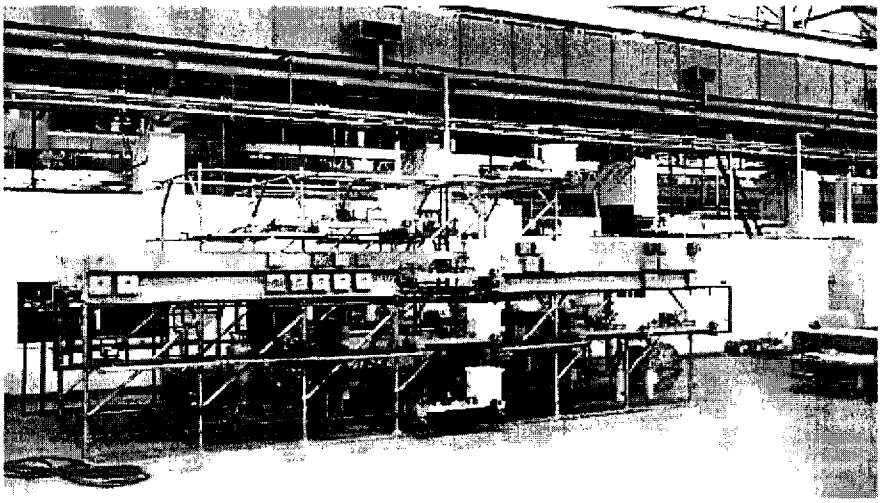


Figure 58: Chilled Water Reduced Scale Advanced Demonstrator.

6.2.1 Integration

The model integration was done using Phoenix Integration's ModelCenter. ModelCenter is an environment designed to easily interface different contributing analyses (CA). It makes extensive use of its graphical user interface to aid the integrator in the task of integration. The drawback to ModelCenter is that it was not designed to integrate time-domain simulations and the process had to be modified to be able to operate in manner desired. Two portions of the integration were crucial, ensuring data synchronization and flexible scheduling. Synchronization can be a problem when large number of CAs are sharing data at disparate times, prompting them to use data from the incorrect time step. Flexible scheduling is necessary to optimize the runtime of the environment, allowing for the execution of those CAs with smaller time constants to be executed more often, and vice versa.

Figure 55 is a depiction of the final IEP model integrated in ModelCenter, including the variables that were transmitted between the modules. The environment transmits at each time step all of its information to a repository that stores all interdependent states and commands. From this central repository, the information is disseminated to the required analyses. This ensures that the model is synchronized and the time-domain data exchange is done sequentially. The embedded Scheduler function in ModelCenter allowed the CAs to be executed in different orders and frequencies.

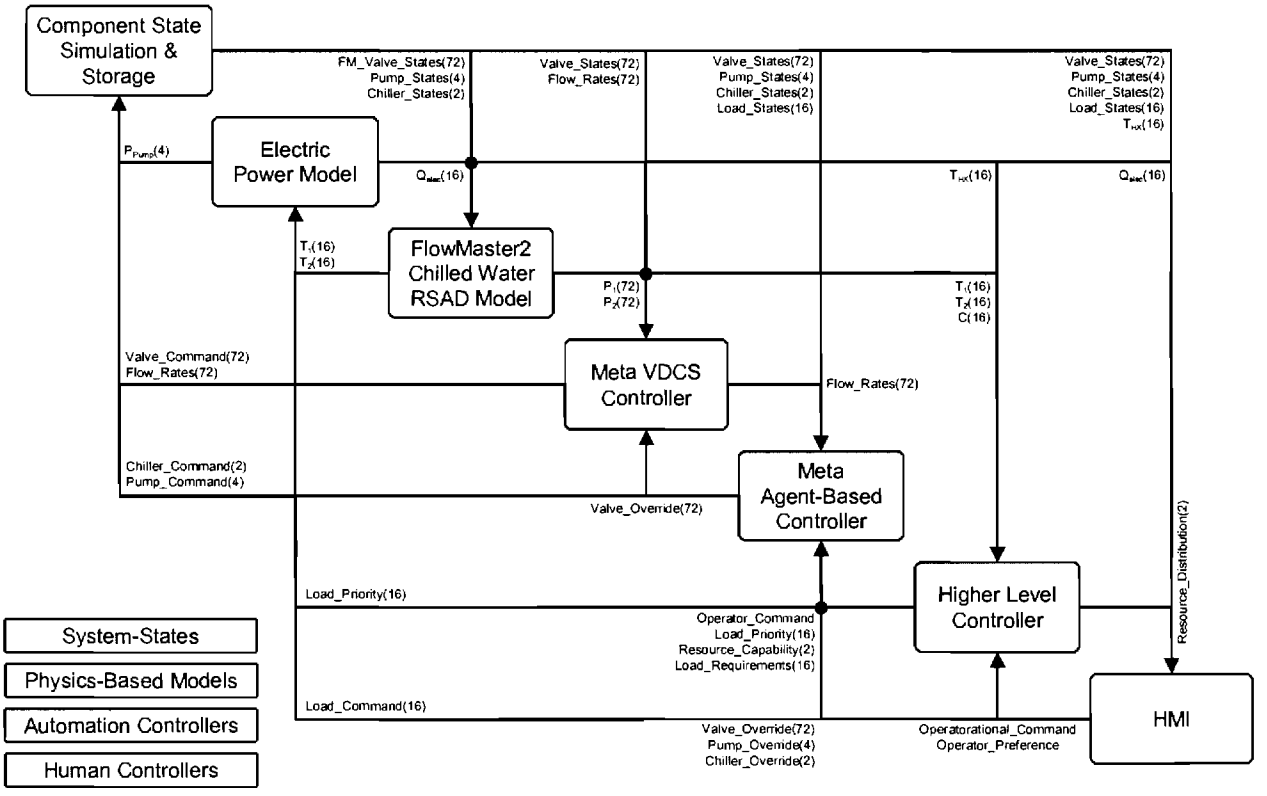


Figure 59: Integrated IRIS Model.

6.2.2 Physical Models

The physical models describe the physics of the electrical and fluid systems. These were developed from models provided to the research team by other institutions and private companies.

6.2.2.1 Low Fidelity Electrical Model

As mentioned earlier in this report, neither the NCS nor the ZELDA models were adequate for this research effort. ZELDA was more suitable as a starting point for a low fidelity model while NCS serves well as basis for a high fidelity model.

ZELDA's architecture was its strong point so it was studied in depth. Basically, ZELDA is a model of the power distribution network on a ship. It has two main power buses; port and starboard. Power is allocated to units referred to as loads according to this load's priority. Several loads are grouped together in one collection module, and a set of these modules form a zone. Zones in ZELDA are spatial zones, i.e. they refer to an area or a region on the ship. Each load has a set of priorities depending on the operational condition of the ship. Power is allocated to each load depending on its priority, and once all available power is allocated to loads with higher priorities, the logic in the model sheds the remaining loads. Neither independency nor networking is modeled in ZELDA. The conclusion was to use this high level architecture in a low fidelity model, substituting the power system components sub-models with more appropriate ones that account for both interdependency and prioritization, and that have a minimum level a of a physics based model. This development is in two stages.

Stage One Development

The ASDL team took ZELDA apart, studied it thoroughly all the way to its seventh level and to the simplest building block. Following this, the team started programming new components from scratch, taking into consideration the model requirements referred to in the methodologies section. The result was a library of six main modules as follows:

1. Resource Generation Module: In this case, this module approximates a constant power source. A random or sinusoidal component can be added easily.
2. Load Module: The load module represents any power consumption component on a ship, regardless whether it is an AC or DC component. The load has an identification number for reference and for generating reports, has a demanded power rating, an available power property, and three switches, namely a control switch (to link to the external control system), an on/off switch (to give the ability for a user to turn the load on/off) and damage switch (to disable the load if flagged to be nonfunctional by an external damage model). Most important, the load has a priority property that sets its importance in comparison to other loads.

3. Zone Module: Five load modules are grouped together in what is referred to as a *Zone*. The zone divides the power required by each load into two parts, one sent on each bus. This required resource is processed in the control module, together with the priorities, and the available power for this load is sent back on the bus.
4. Node Module: It is more like a T-section in a pipe. It takes in the total demanded power form the zone and adds it to the bus. It also distributes the available power from the bus to the zone.
5. Connector Module: It connects two nodes. The collection of the connector modules and node modules forms the bus. One thing about the connection module that a node does not have are the properties damage and control switches. Hence, the bus can be disabled by turning off the damage switch (if the damage models computes that the bus is nonfunctional) or by turning off the control switch if the control algorithm decides to disconnect the power to this particular zone.
6. Priority Algorithm Control Module: This module is the heart of the model. It allocates the resource to the loads, based on their demand and their priorities. The higher the priority a load has the better chance it is supplied with the power it demands. Priorities are set in two levels, so that when two loads have the same priority, the second level is considered. The inputs to this module are the power demand from each bus, the priorities, and the outputs are the power assigned to each load, on each bus.

The team chose Simulink to be the modeling tool at this stage. No load interdependency is modeled here, nor the actual physics of the electrical system. The objective was to replicate ZELDA with a modified version that is in a more useful form and the team was successful in accomplishing this. The Stage One model is shown in Figure 60. The model has two zones (shown in gold) in contrast to 7 zones in ZELDA. Each zone has 5 loads connected in parallel. Similar to ZELDA, the model has a port bus (blue) and starboard bus (orange). As previously mentioned the bus is composed of node modules which link to zones, and connector modules which connect nodes to each other. Adding new zones, i.e. expanding the model, only involves drag and drop of a zone, two connectors and two nodes, and setting up their parameters. A very easy task if one compares it to ZELDA's rigid architecture. The power allocation module is shown in pink. It receives power requests and priorities from loads, processes this data, and assigns power to each load, then sends this power assignment matrix on each bus. Power is divided between the two buses depending on the total power request on each bus.

Due to the use of Simulink (which is more of a differential equation integration environment, in a graphical form), the model ran into algebraic loop difficulties. To correct these errors, memory modules had to be added to the architecture to store previous values, and tedious initialization of all signals was required.

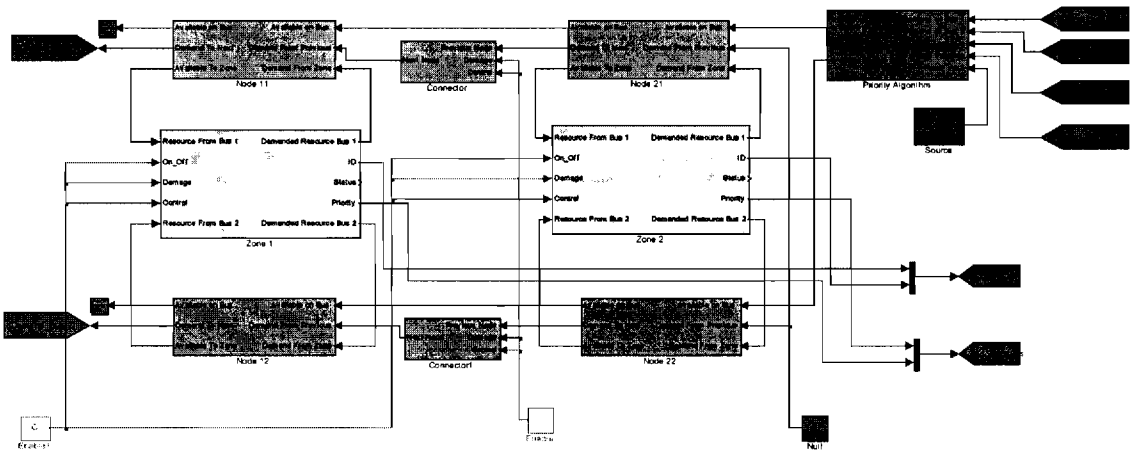


Figure 60: Stage One Power System Model

Stage Two Development

The objective of the second stage is to avoid the pitfalls of the first and at the same time model the simple physics and interdependency. The team decided to model the interdependency network and the physics based heat dissipation in Simulink, and use MATLAB™ scripts to compute the power allocation. This architecture integrates easier in the full modeling and simulation environment.

The inputs to the Stage Two model are the temperatures of the cooling fluid from the flow model, and the load priorities from the control model. The cooling fluid temperatures are used to calculate the internal temperature of the component. If this temperature increases beyond a given threshold, the component fails. Load priorities are determined by the control algorithm in a different model. These priorities are assigned to different components such that the ones with higher priorities are served first. The output from the model is the state of each component, its power consumption, and heat loss if any. A simple physics based heat loss model is used here. Figure 61 is a schematic of the model.

For every time step, the model does two things. First it figures out the power supplied to every load using the prioritization algorithm. As mentioned, a complex set of MATLAB™ scripts does that. It starts with an initialization of component states, and then applying the priorities to components. An interconnectivity matrix is checked for components that are not connected to a viable power route and these components are shed. The power required by each component is then updated accordingly (which is different from the power rating). For example, if a load has a power rating of 200 Watts, but is not connected to a faulty converter rendering no viable power route to it, then the power required is 0 Watts. The result is a set of intermediate states which are fed to the interdependency Simulink part of the model.

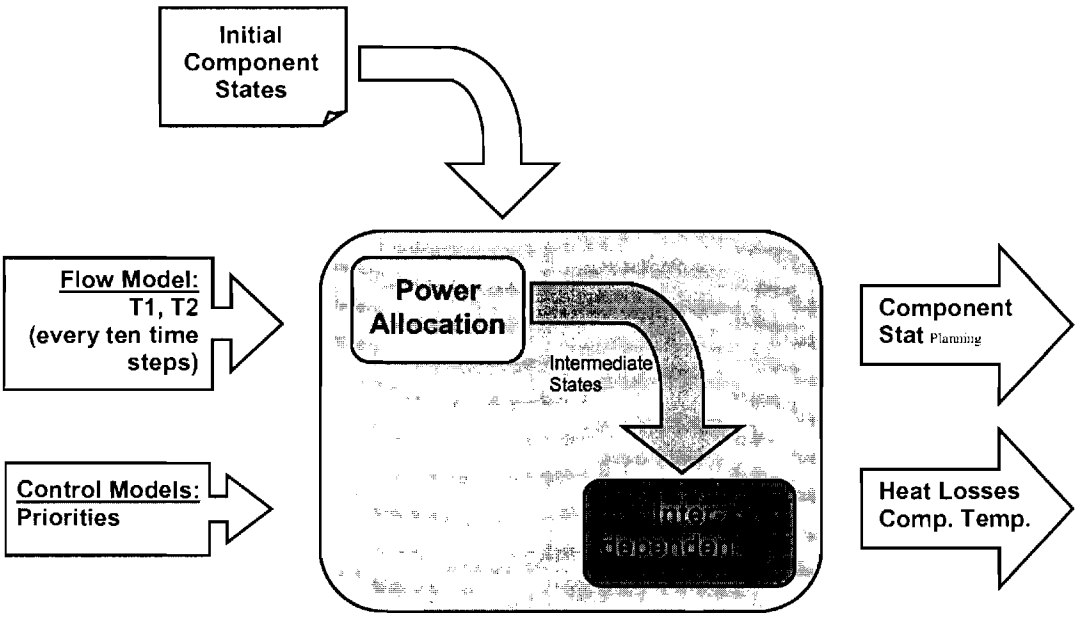


Figure 61: Stage Two Power System Model

The interdependence/physics based Simulink model is shown in Figure 62. Similar to stage one, the team built a new library with new modules, which included a load module, a pump module a converter module and several data routing and acquisition modules. Basically, converters are components that pass through electric power with some heat losses. The model computes the losses of each converter based on a constant efficiency related to this converter. In a higher fidelity model, logic can be added to the converter component that will make this efficiency change with the converter parameters.

Most loads link to a converter but not to the network directly. Shutting down a converter shuts down all the loads (or even converters) downstream. This is a very important feature that will prove useful when a damage model is added to the simulation environment. It also simplifies the control algorithm. For example, if each converter is linked to a specific zone, shutting down this converter sheds all loads in this zone. Hence, control can be applied both on load by load basis, and load groups basis.

Loads are similar to the load components in the Stage One model. They produce heat due to power loss, and are in constant need of cooling. Similar to converters, a load has a constant efficiency that determines the heat loss. The designer can connect a load to one converter, several converters for redundancy or even directly connect to the power source. The Simulink graphical user interface facilitates making these connections, and almost any logic or network can be implemented. This includes dynamic networks in which the network architecture can change real time by applying the appropriate logic.

To cool the converters, pumps are required. Pumps consume power, but do not produce heat (in the ideal representation). The logic in the model is set such that if a pump cooling a converter fails the converter keeps on heating up to the point at which it reaches a threshold and shuts down.

Figure 62 is an implementation of the interdependency model, with 15 loads, 4 converters and 4 pumps. Loads are shown in gold, converters in cyan and pumps in green. All the other components are either data acquisition or data routing modules. These are only necessary for proper operation of the model, but do no represent real components. It is the assumption in the low fidelity model that the electric engineers well designed the network and the components on it, such that we, as system integrators, are not concerned with the choice of cables, or switches, etc. at this stage.

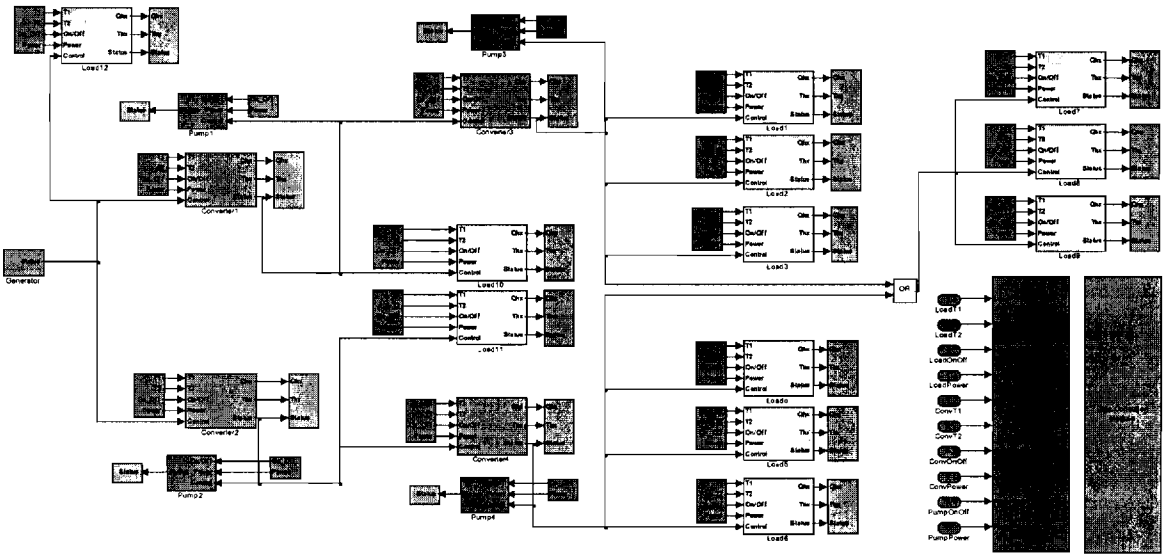


Figure 62: Stage Two Simulink Representation of the Power System Model

The Stage Two model has all the characteristics of the needed low fidelity model. The researchers used this model in the modeling and simulation environment to represent a power model. It was easy to integrate in the environment, is easily expandable and has an acceptable level of physics based modeling for a low fidelity requirement. It runs fast enough to keep up with the fastest sub-models in the modeling and simulation environment. Nevertheless, a higher fidelity model is needed to accurately simulate the actual power loss based on the load states, not based on a constant efficiency. Transients representation in the low fidelity model is very simple (a fist order lag) due to heat transfer, while in reality, they should calculated from the solution of a set of differential equation. A higher fidelity model is essential for different reasons such as fine tuning the design and detection of limits that might arise from internal peaks in electric components. The higher fidelity model is discussed next.

6.2.2.2 High-Fidelity Electrical Model

Currently, a power flow model, based on a modified ZELDA architecture is being used as the integrated electrical model for the IRIS environment. A modified NCS electrical model will be available in the near future for facilitating higher fidelity calculations. Eventually, the possibility of combining the two modified models (NCS and ZELDA) into one unique electrical model will be examined, in order to include the advantages of both models. A combined model will be expected to have the electrical component architecture and the rich configuration (number of electrical components and detail in their connectivity) of the power model, added to the depth that the physics-based capability at lower levels (physics-based dynamic calculations inside of every component) of the NCS model can provide

ASDL has developed simplified and modified versions of existing tools in lieu of an electrical model, assuring their compatibility for integration purposes. However, ASDL's main role is less to create and develop simulation models and more to be able to integrate stand-alone models that are obtained by research partners. The reason why ASDL is involved in modifying existing simulation models is mainly for the purpose of testing and developing the IRIS integrated environment and making sure that it will be capable of accepting and incorporating seamlessly any simulation model available to be integrated.

As mentioned previously, a sophisticated electrical model is being developed by ASDL for future use in the IRIS framework, based on a simplified version of the Purdue NCS electrical model. Its simplified architecture is described in the following figure.

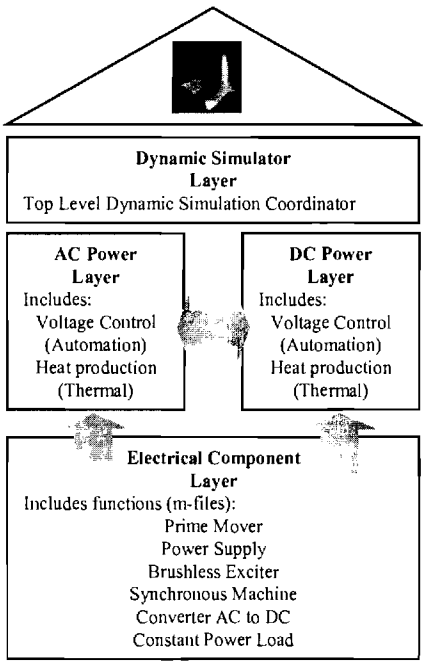


Figure 63: Simplified NCS electrical model for integration in the IRIS M&S environment

The simplified NCS model is implemented in MATLAB™ for ease of manipulation and integration. Therefore, it can be more compatible to the "wrapping" environment and it is much easier for the user to perform changes in the source code. Additionally, such framework will allow for modifications of component parameters and initial conditions, even during the simulation. The same exact physics equations are being used, extracted from the source code. However, in this simplified version, the original level of sophistication is only maintained for AC and DC power generation.

The conversion from ACSL™ to MATLAB™ required thorough theoretical analysis and understanding of the original source code. The next step was to break the source code down into smaller modules. Extensive understanding of every module took place in order to allow for conversion of every module to a MATLAB™ m-file, thus creating a function necessary for building the higher level layers. For building the different layers, the modules were recomposed, based on a simplified architecture, as shown in the figure above.

Concerning the status of this process, the model currently model is built and has been verified. Future tasks until full completion of the effort will include calibration of initial conditions and parameter definition based on original model, as well as model validation for obtaining results that actually make physical sense. The simplified NCS model is expected to be integrated into the IRIS environment in the near future.

6.2.2.3 Chilled Water Model

Valves

The 72 valves in the CW-RSAD must communicate with their agent-level controllers in order to be classified as "smart" valves. In the integrated simulation environment, both the CW-RSAD model and the agent-level controller are realized as modules, called plug-ins in the ModelCenter framework. ModelCenter provides its own Application Programming Interface (API) to allow users to build their own custom plug-ins that are able to communicate with other plug-ins and ModelCenter with script languages supported by the framework. The ModelCenter API is used for connecting module-to-module or module-to-ModelCenter, while the Component Object Model (COM) object is used for connecting between an application and the designated module that it represents. Flowmaster2 allows a user to generate the COM objects by adding special components called COM-enabled components in its GUI model design environment. After putting COM-enabled components in the GUI environment, the users can make the instances of those COM objects in the code. Figure 64 describes how the FlowMaster2 model and the ModelCenter framework communicate each other.

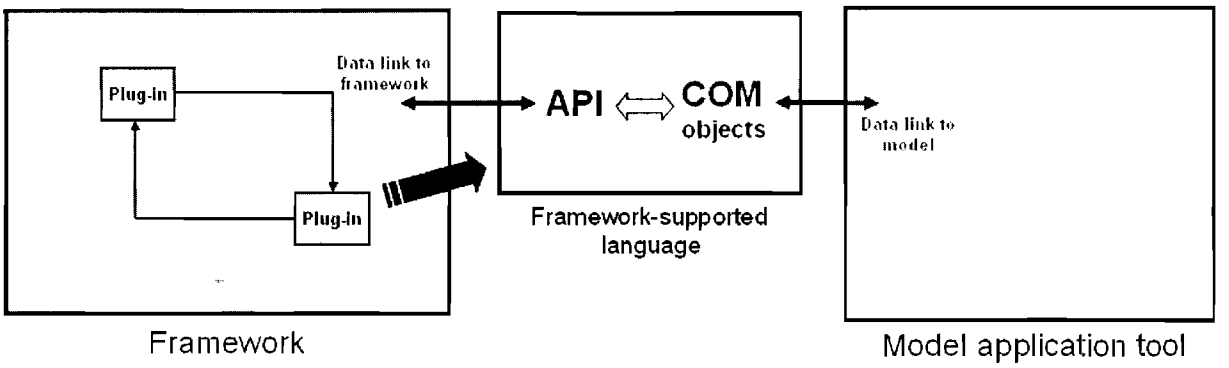


Figure 64: Plug-in data connectivity

This approach is also applicable to other interconnections existing among subsystems. The thermal flow input to the heat exchangers, pump rotational speed values, and the outlet temperature of the two main chillers were able to be read from outside using the same technique.

Thermal analysis capability

One of the major and direct interactions with other systems found in the fluid system is the heat exchange with electric or mechanical components. The amount of chilled water needed by a certain service load was predicted and decided based on the heat energy that should be removed for the related component or device to be operated safely. In the modified chilled water system, each service load receives the value of thermal flow rate as an input to calculate the outlet temperature of the heat exchanger in a service load. The thermal flow rate is estimated from the temperature difference between the component in contact with the heat exchanger and the average water temperature at the inlet and outlet of the heat exchanger. That means the computation should proceed in an iterative manner in a discrete time sequence. As a heat producing component loses heat and is cooled to a desired temperature, an agent-level controller will reduce the water flow coming into the service load network by decreasing the valve opening so that the temperature is maintained in the safe operation range. All this setting for estimating the thermal flow rate is not necessary in a real situation. In the real physics, the agent-level controller only needs to monitor the temperature of the component being cooled and send a feedback signal to the valve to regulate the component temperature. The temperature sensing for the leaving and entering chilled water at the heat exchanger is purely a setting needed in the software model to estimate the thermal flow rate, so those temperature sensors might be referred to as virtual temperature sensors that don't exist in the real world. Figure 65 is the description of the heat transfer computation scheme that was added on each service load network in CW-RSAD.

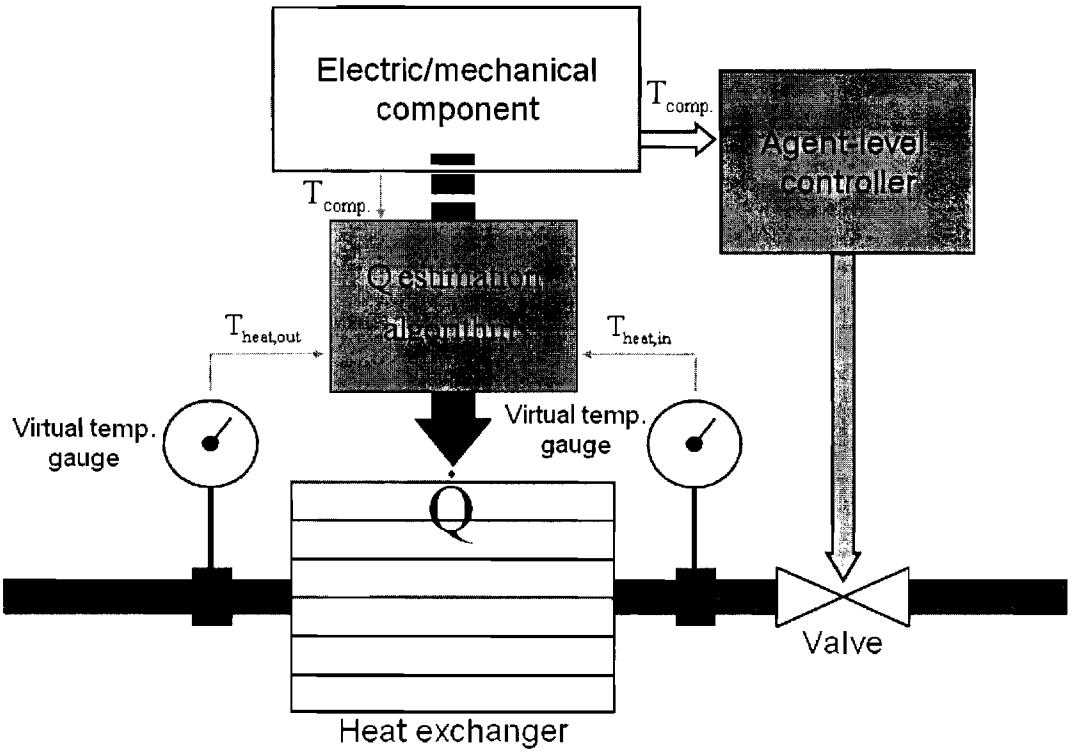


Figure 65: Software expression of thermal activity at a heat exchanger

Pump control interface

This is another interaction made between the chilled water system and the electric power system because the pumps need both higher order controls and electric power from the power system. In the view of the power system, the pumps in the chilled water system are one of constant voltage loads. Typically, the rotational speed of the pump is proportional to the electric current provided to the pump so the higher order controls can control the pump speed by varying the current supply. By this reason, the pump RPM was selected as the control input variable. Figure 66 shows how the pump rotational speed control is implemented.

Two main chillers

Basically, the chiller units have the same physics as the heat exchangers in the service load network but the only difference is that the heat energy of the chilled water network is dumped out through the chiller. This chiller originally transfers the heat energy to the lower temperature sea water system so the same thermal analysis scheme can be used to describe the service loads. However, the sea water system is not available so an alternative way of simulation for the heat transfer at the chiller is needed. Therefore, the outlet temperature of the water flow leaving the chiller was chosen to be the direct input parameter. In such an approach, the observer of the integrated simulation must provide a suitable outlet temperature.

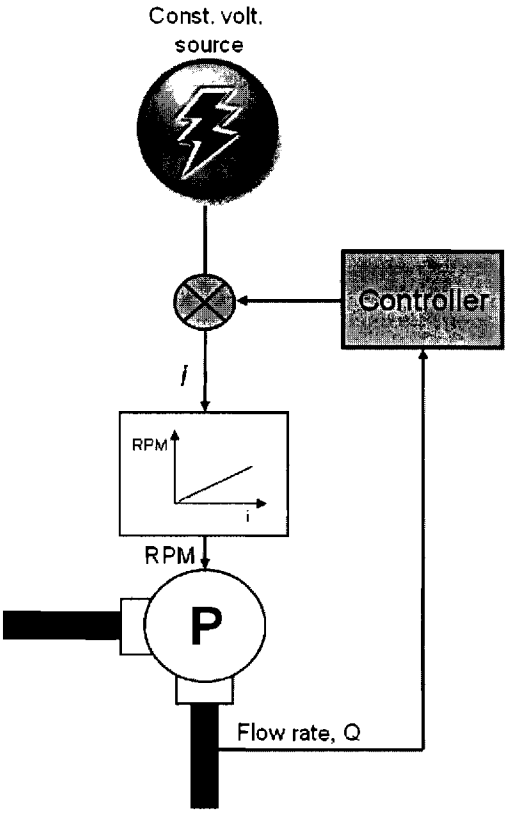


Figure 66: Pump Control

Rupture valves

The current CW-RSAD software model simulates the rupture by putting valves on some pipes that can possibly be damaged in a certain damage scenario in mind. This approach is very inflexible and inconvenient when it comes to perform rupture analysis for multiple damage scenarios. Also, the assumption that the fully or partly opened rupture valve can act similarly like the real rupture was not validated yet. In spite of all those problems, the reason that the builder of the CW-RSAD software model had to choose this approach seemed to be simply because the Flowmaster2 doesn't provide the capability of rupture analysis. At this moment, the original approach of rupture analysis was used so that the modification was just the addition of the capability of changing rupture valve opening from outside the model using the same way explained in the section for smart valves.

Nomenclature of interactive components

As the number of components who need to communicate with other subsystems during simulation grows, it became more important to maintain expandability in the future modifications or applications and handle a large number of input-output interfaces among the plug-ins in the framework. To manage this more efficiently and flexibly, names were given to those interactive components in a systematic manner.

All the interactive components have the COM-enabled components to communicate with other subsystems and also have names by which the COM objects are called in the code. Figure 67 is the example of naming for a valve component in the chiller network of CW-RSAD.

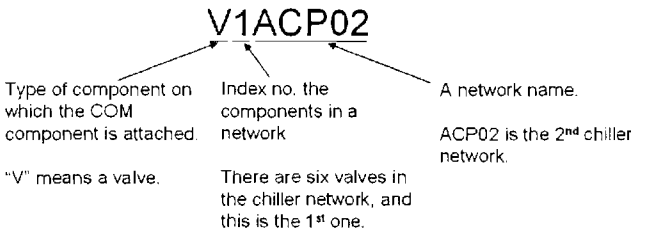


Figure 67: Example of interactive component naming

The systematic naming additionally facilitates building the fluid model plug-ins. Automation was achieved by building a fluid model plug-in and the scheme is described in Figure 68. Using the automated plug-in, one can easily build a fluid model plug-in without writing any code. Once the other fluid subsystems are provided, they will be added to the integrated simulation environment with great ease.

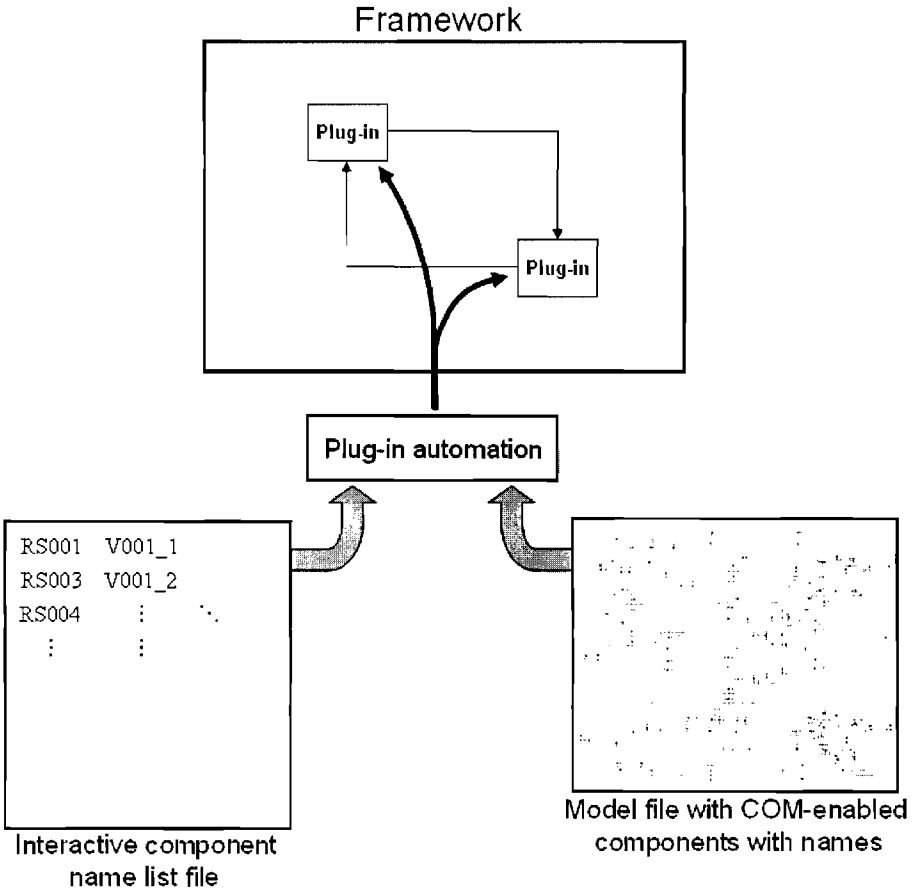


Figure 68: Plug-in Automation

6.2.3 Control

To achieve the goal of survivable distributed control, the IRIS team has identified the Open Kernel Architecture Agents developed by at the Applied Physics Laboratory at John Hopkins University [Schiedt, 2002], the Virtual Distributed Control System developed by Fairmount Automation and Drexel University [Kam et al., 2004], and the Strategic Protection Infrastructure Defense Multi-Agent System (SPIDMAS) a control architecture for power distribution systems developed at the University of Washington [Jung and Liu, 2001]. Using these three systems as inspiration, the IRIS team has envisioned the IEP control architecture to be a hierarchical, distributed, easy to adapt and expand system as depicted in Figure 10.

The control architecture discussed previously has been preliminarily integrated into the M&S environment. Figure 59 depicts the integrated environment, in which the controllers are the labeled metaVDCS, ABCtrl and HLCtrl. Each one is a MATLAB™ script that attempts to mirror the functions of VDCS, the OAK agents and the high-level controllers respectively.

The metaVDCS script attempts to model the logic in the smart valve controllers. The goal is to contain pipe ruptures by measuring the pressure gradient through the valve and estimating the flow rate. If the change in flow rate exceeds a certain threshold, the controller assumes that there is a rupture and commands the valve to close. The valves can also gradually open once the rupture has been contained.

Overseeing metaVDCS is the Agent-Based Control System (ABCS) labeled ABCtrl. The ABCS uses more complex logic to regulate the flow through the different loads according to their priorities obtained from the High Level Controller (HLC). The ABCS also controls the pumps and chillers, and regulates their use as cooling is required by the service loads.

The HLC at this stage serves as a simplified prioritization algorithm, assigning the priorities to each service load according to user inputs provided by the Human Machine Interface (HMI). Future work will involve incorporating a Markov Decision Making Process to account for the uncertainty in the state of the system.

The desire to optimize manning and the functions of the crew requires that the ship be autonomously reconfigurable, but it is essential that operators have the ability to override the decision making systems. For this reason the IRIS team has been devoting considerable resources and time to the development of a Human Machine Interface (HMI) that allows operators to supervise and interact with the system. The HMI framework is being developed using Asynchronous JavaScript and XML (AJAX) to enable the quick development and integration of interfaces to the IEP environment. The component labeled "HMI" in Figure 7 serves to send the data and receive commands from the HMI. The HMI has been crucial in debugging the system and understanding the behaviors of the IEP and its controllers.

In order to integrate the M&S environment, the IRIS team built the Complex Systems Modeling and Simulation (CSM&S) Environment, which uses a dual processor, quad-monitor computational station that allows for the simultaneous visualization of large amounts of data. Figure 69 depicts the how this system was used by the students.

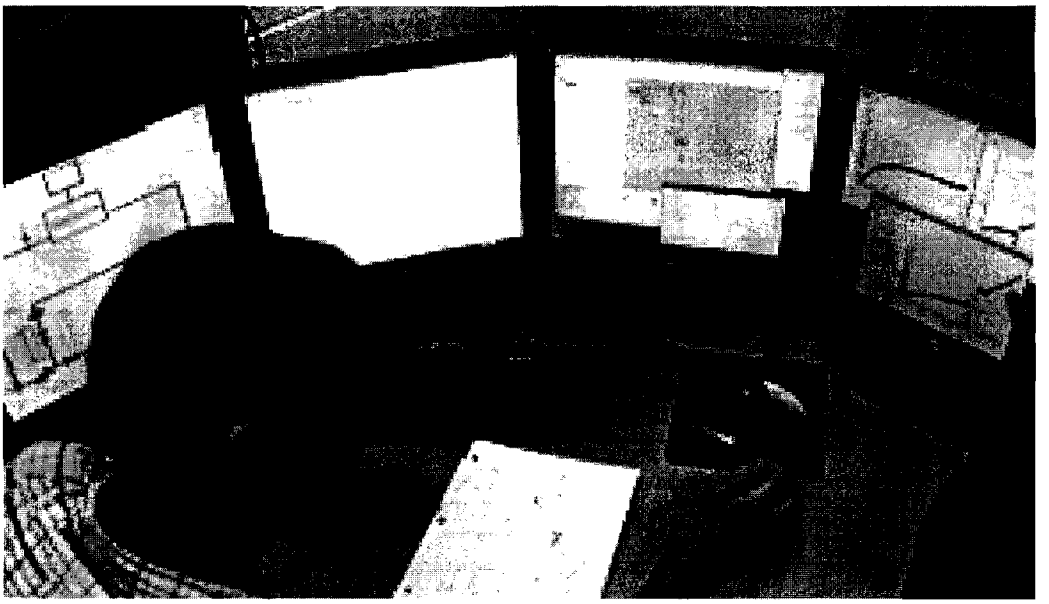


Figure 69: Complex Systems M&S Environment

Including aspects of cognitive engineering and human factors, the IRIS team hopes to help bring the development of HMIs earlier into the design process in order to allow designers and automation experts to better understand what the limitations of human-in-the-loop control are. The need to optimize manning demands that the automation engineers understand what the Navy wants and what the operators can do.

6.2.3.1 Agent-based Control

Establish Agent-Based Control for Ship Chilled Water System

The goal of this step is to develop an agent-based control system for the ship chilled water system, namely the Chilled Water Reduced Scale Advance Demonstrator software model.

Decomposition

From Figure 70, the whole RSAD model is divided into three subsystems which are Chilled Water Resource subsystem, Cross Valves Subsystems and Service Load Subsystem respectively. Resource subsystem can be divided into two relative independent resources further. Each resource includes two pumps, one chiller, one reservoir and several valves. The Cross Valve Subsystems can be divided into two groups, one groups includes valves in the route for the service loads getting chilled water from

resources and the other include valves in the route for the service loads sending used water back to the resources. The service load can be divided into 16 relative independent service loads. Each service load includes one component which needs to be cooled down and several valves. There are many flowmeters which play the role of sensors to gather information from the model.

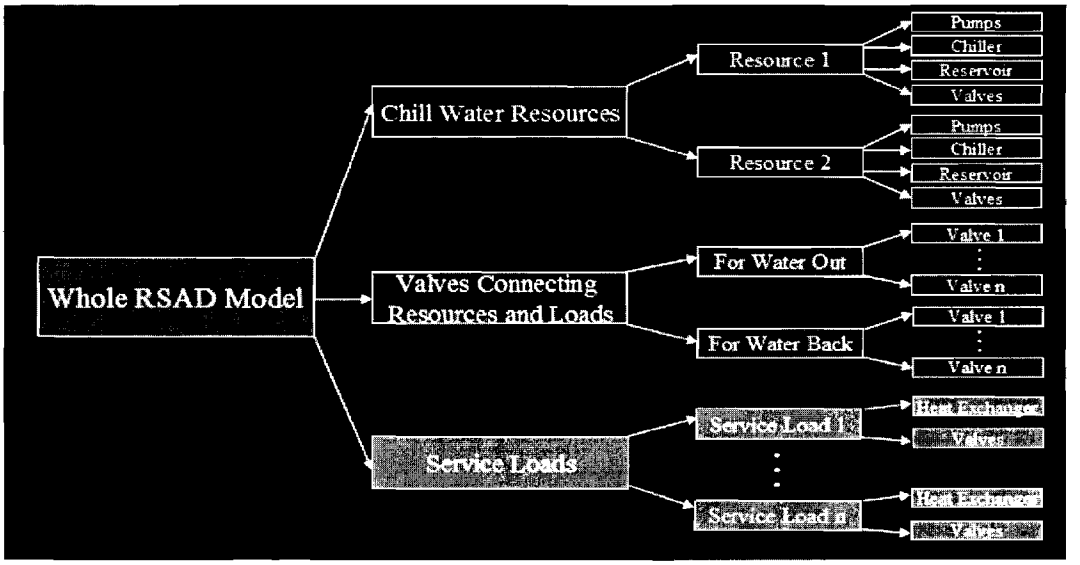


Figure 70: Decomposition of CW-RSAD Model

Abstraction

After decomposition of the whole system, the next step is to analyze the structure for each portion and design a reasonable internal logic to control it and define the inputs and outputs for it.

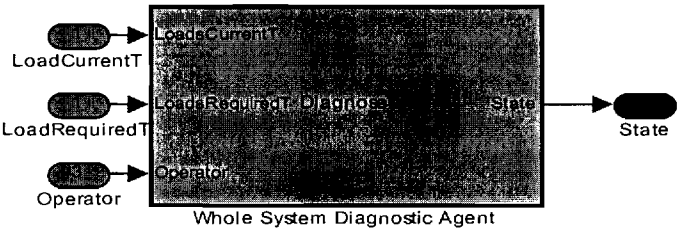


Figure 71: The Whole CW-RSAD Model Diagnostic Agent

The Whole RSAD Model Diagnostic Agent is a simple agent which detects the state for the whole system according some information from sensors. It includes 3 inputs and one output: Service Loads Current Temperature which is a 16*1 vector, Service Loads Required Operating Temperature which could be fixed or could be changed according different situations; Operator which is a command from the operator. The state is a scale which is the whole system state. The internal logic for this agent is: once the

system detects one service load whose current operating temperature exceeds its required operating temperature, the system state will become abnormal.

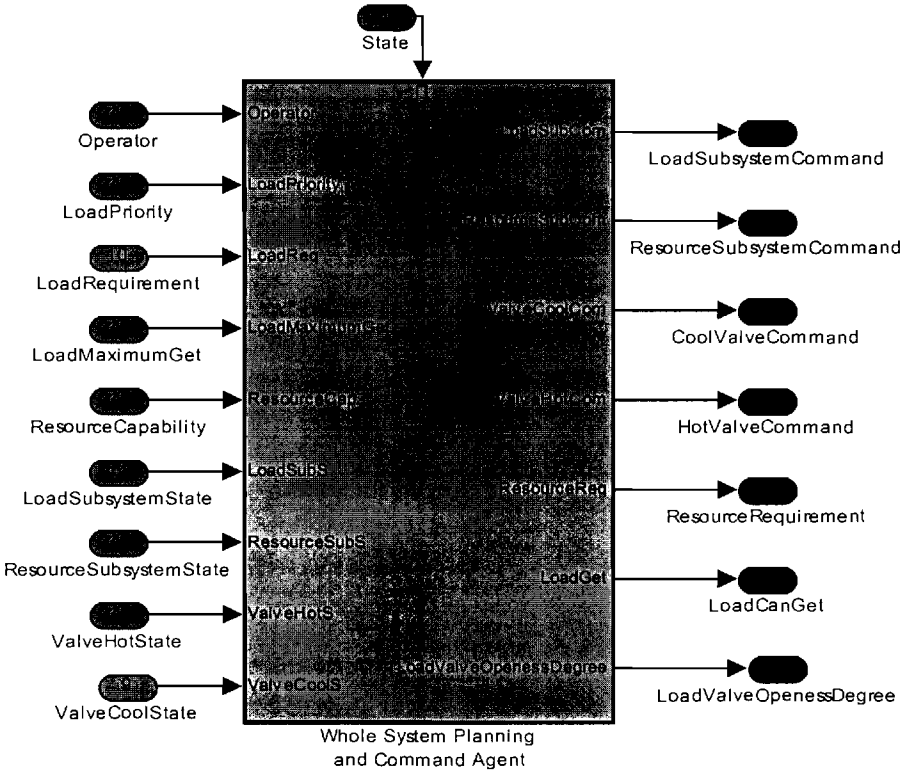


Figure 72: The Whole System Planning & Command Agent

The Whole System Planning & Command Agent is in charge of the whole system and takes each subsystem as an integral entity and does not care about any detail information for each subsystem. It has 10 inputs and 7 outputs. The first input is the whole system state. If the whole system state is normal which means everything is working, so the system do not need to give any planning & command, just leave the system the way it was.

Load Priority, load required quantity of chilled water and resource capability is from the higher level (ship level agent). Resource subsystem states, service load subsystem state and Cross Valve State information from the component level. The outputs are: the commands to each service load, the commands to each chilled water resource, the command to the cross valves and the quantities of required chilled water from each chilled water resource, the quantity of chilled water each service load could get and the openness degree for each valve in service loads which is relatively proportional with the quantity of chilled water each service load can get in the current situation.

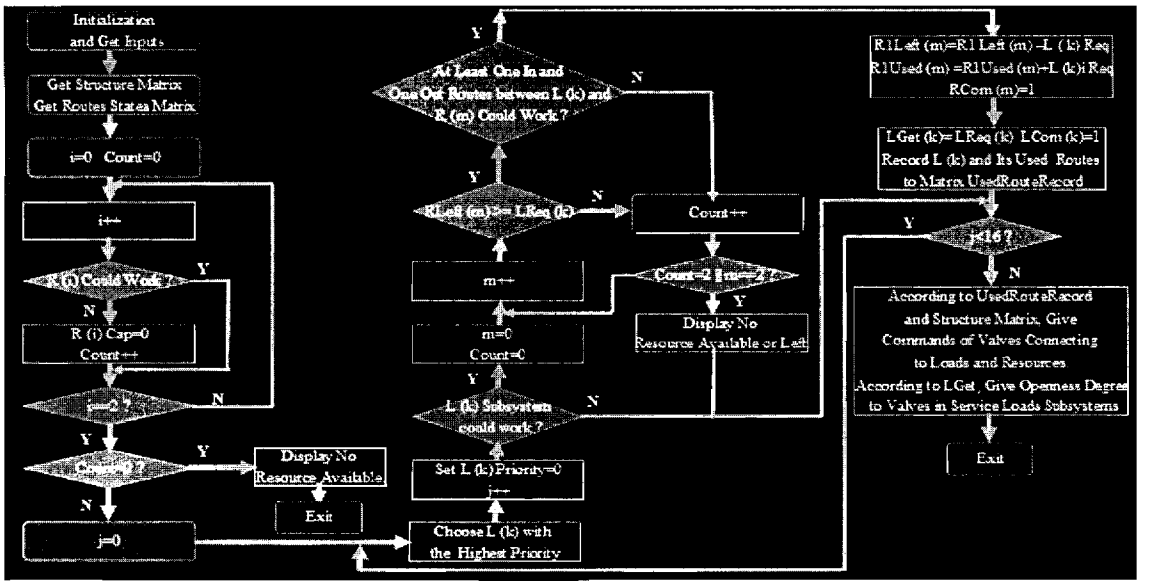


Figure 73: Internal Logic Flow Chart for the Whole System Planning & Command Agent

This internal logic for the Whole System Planning & Command Agent is a little bit complex. It is used to choose which service loads to get chilled water from which resource by which route. Firstly, the structure of the whole system information is stored in a bunch of matrixes. It will consider each service loads in the order of their priority. Firstly, it check the state of the service load with the highest priority, if the service load state is damaged, which means this service load can not get any chilled water, so it will check the service load with the next highest priority. If the service loads state is good, then it will check resource 1, if resource 1 is damaged, it will check resource 2. If resource 1 state is good, it will check the state of the cross valves which connect this service load and resource 1. If none of the routes is working, which means this service load can not get water from resource 1, so it will check resource 2. If resource 2 is not working or none of the routes between this service load and resource 2 is working which means the service load can not get any water from any resource, so it will distribute zero resource to this service load.

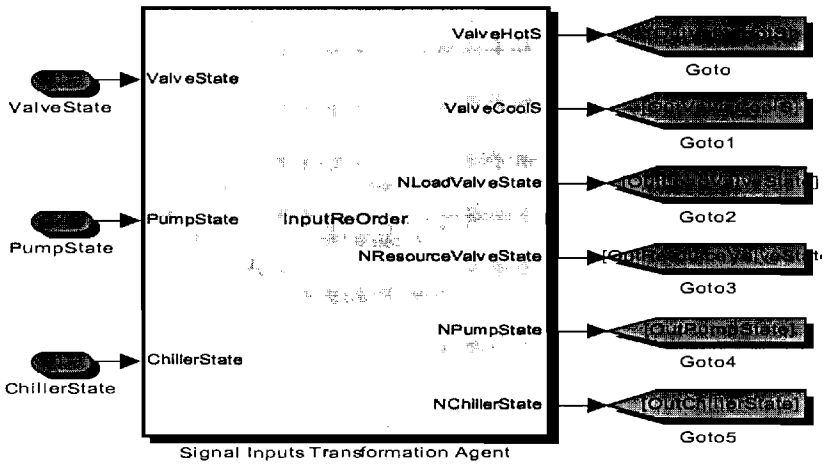


Figure 74: Signal Inputs Transformation Agent

Signal Inputs Transformation Agent is used to rearrange the inputs which will be sent to individual agents. For example, in the whole system, there are 72 valves which is a big vector, but for the resource, it just needs 6 valves information to be drawn from the big vectors. By using the Signal Inputs Transformation Agent, it makes the interface for other modules in IEP much friendlier.

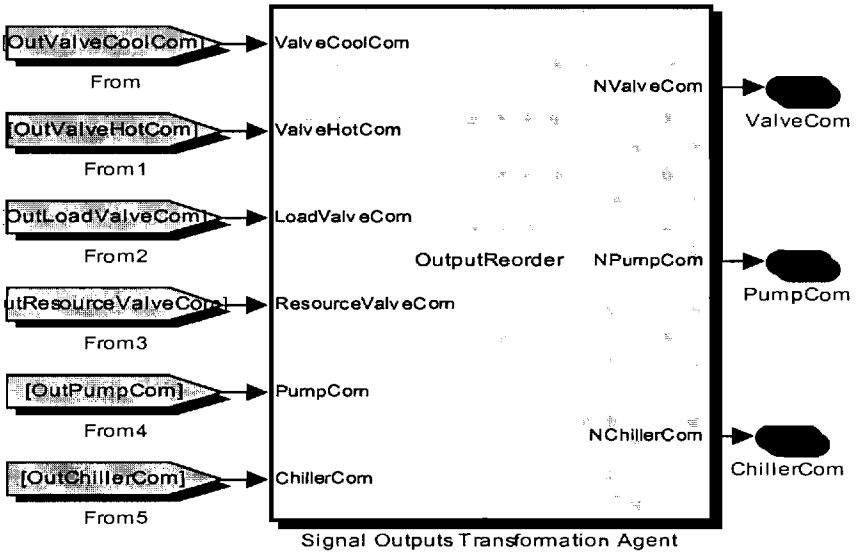


Figure 75: Signal Outputs Transformation Agent

Signal Output Transformation Agent is used to deal with the information gathered from different agents which is need to be send out to other modules in IEP. It is role is similar to the Signal Inputs Transformation Agent and makes the interface for other modules in IEP much friendlier.

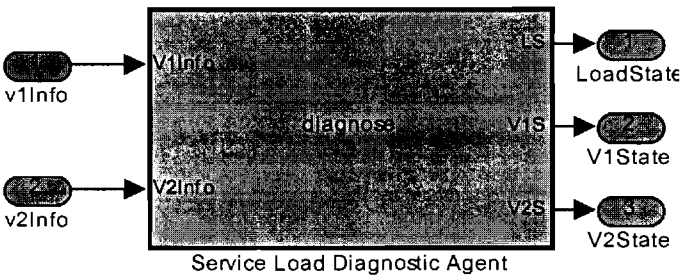


Figure 76: Service Load Diagnostic Agent

There are 16 service loads in the whole RSAD model. Each service load includes one diagnostic agent and one planning & command agent. The service load inputs and outputs and internal logic may different according to the actual service load structures. From Figure 76, the Service Load Diagnostic Agent has two inputs: valve 1 information, valve 2 information and three outputs: the whole service load state, valve 1 state and valve state. From the input information, the diagnostic agent will decide whether the subsystem could work or not.

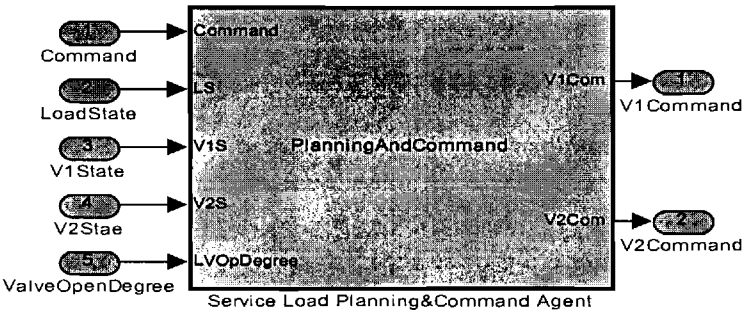


Figure 77: Service Load Planning & Command Agent

The Service Load Planning & Command Agent has five inputs: Command from the higher level agent, the state of this service load, resource requirements (valve openness degree), valve1 state and valve 2 state. It has two outputs: valve 1 command and valve 2 command. When the command from higher level agent is open, the Load Planning & Command Agent will check this service load state firstly, if this state is damaged, which means the service load can not get any resource, so it will give close command to both of the two valves in this service load, if the service load state is good, it will try to give an open command with specific openness degree to one good valve and give a close command to the other valve.

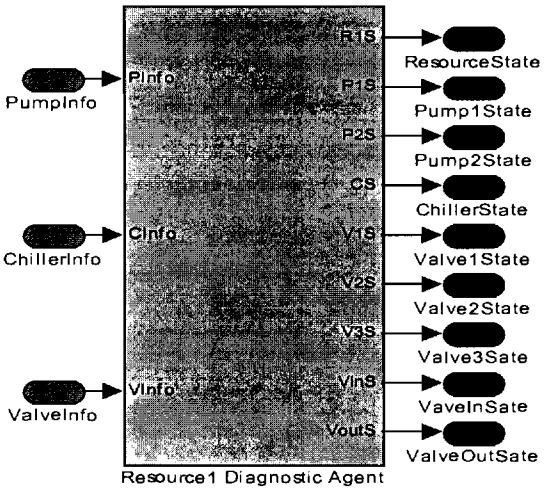


Figure 78: Chilled Water Resource Diagnostic Agent

There are two identical chilled water resources in RSAD model. One of them is redundant. Each resource has one diagnostic agent and one planning & command agent. The diagnostic agent has three inputs: pump information (vector), chiller information (scalar) and valve information (vector) and nine outputs: resource state, pump1 state, pump 2 state, chiller state, valve 1 state, valve 2 state, valve 3 state, valve in state and valve out state. All of the outputs are scalars. The diagnostic role is to decide whether the resource load has the capability to supply chilled water and gives its components states at the same time.

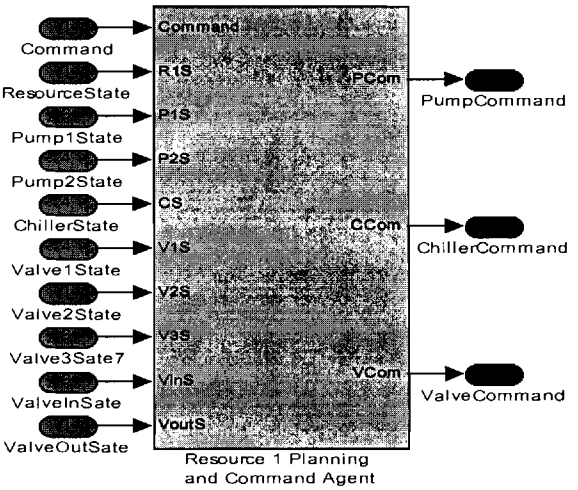


Figure 79: Chilled Water Resource Planning & Command Agent

The Chilled Water Resource Planning & Command Agent has 10 inputs: command from higher level agent, the state of the whole chilled water resource subsystem, pump 1 state, pump 2 state, chiller state,

valve 1 state, valve 2 state, valve 3 state, valve in state, valve out state. All of them are scalars. It has three outputs: pump command (vector), chiller command (scalar) and valve command (vector). This agent role is to give command to its components according to the command from the higher level agent and the state of the whole resource state and its individual components states. For example, when the command from higher level agent is open, which means some service loads need chilled water from this resource, the planning & command agent will check the resource state, if it is damaged, it will give a close command to each components in this resource subsystem; else it will give open commands to the chiller, the appropriate pump and valves and give close command to the other components.

Organization

After the internal logic and outputs and inputs are established for agents, the next step is to connect all of these agents and make them work interactively and smoothly to achieve the system goal.

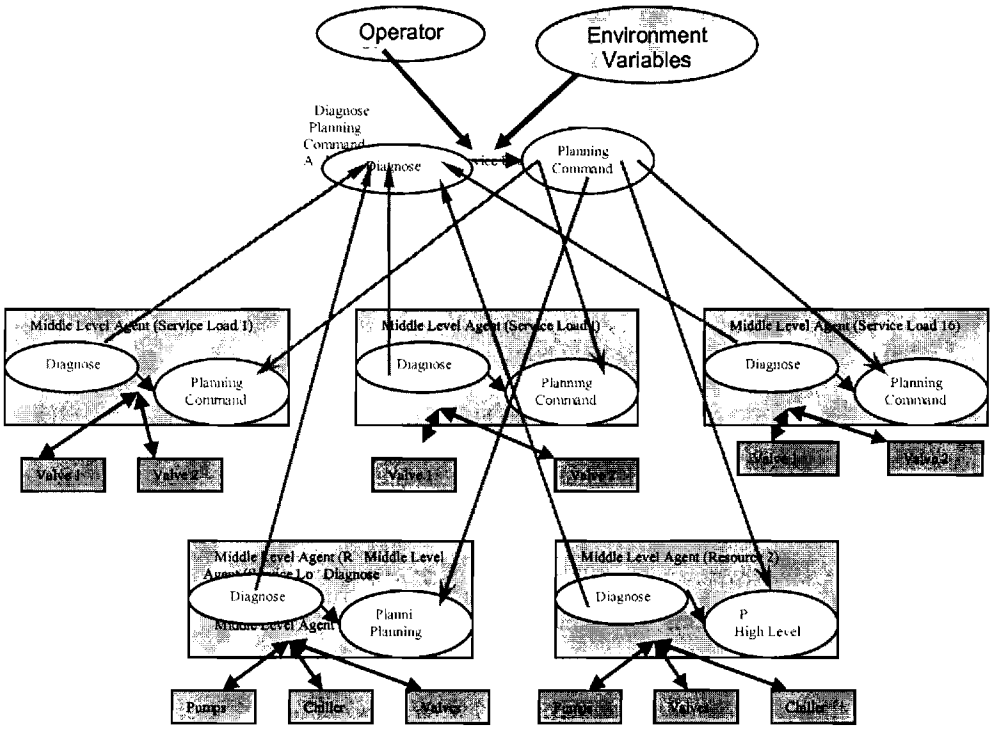


Figure 80: Agent-based Control Structure

Figure 80 gives the general picture of the interconnection between these agents. There are three levels in this structure. The first level is the high level agent which includes one diagnostic agent and one planning & command agent. Do not be confused with high level control in the integrated environment plant (IEP) mentioned before. The high level control in IEP is in charge of the whole ship, which includes power system, chilled water system etc. Here, the higher level agent is just in charge of the whole chilled

water system (RSAD model). The second level is the middle level which includes service load subsystems and chilled water resources. The lower level is the component level, which includes all of individual valves, pumps, chillers etc. Each component is a smart agent which is designed in the metaVDCS which is another part in IEP. Next, more details about the interconnection between agents will be discussed.

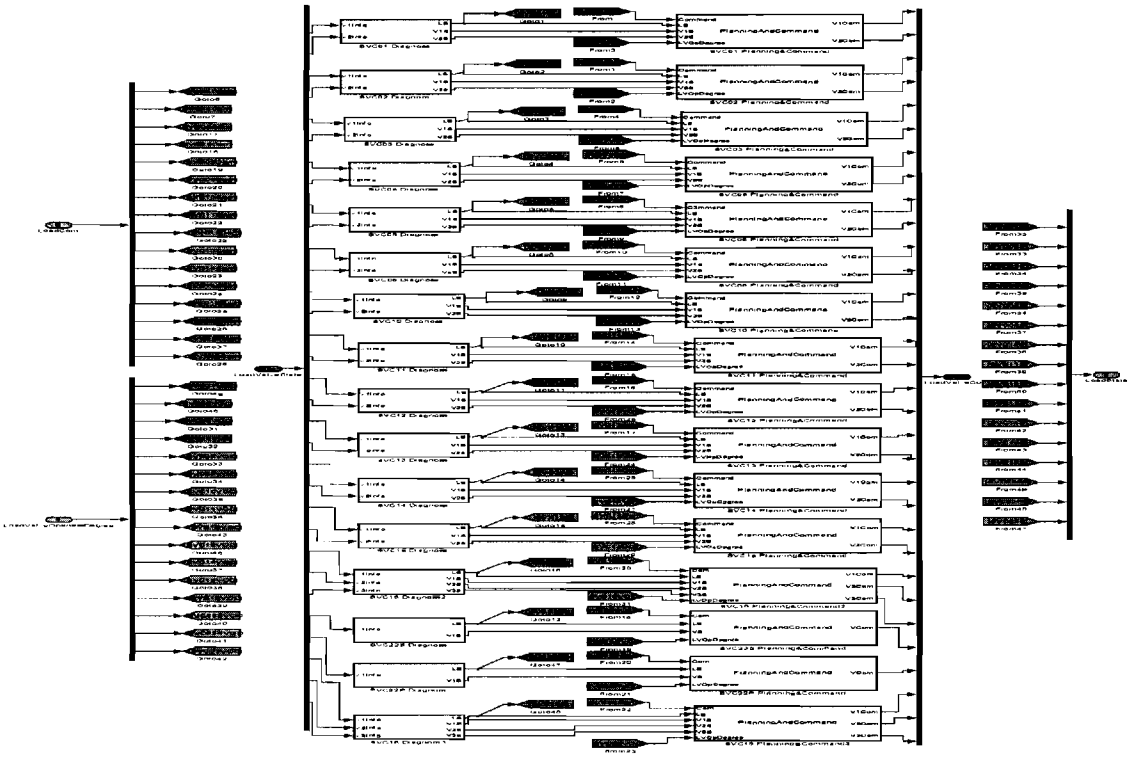


Figure 81: Integration of 16 Service Loads

Figure 81 shows the integration of the 16 service loads. The diagnostic agent and planning & command agent for each service loads have direct relationships. The planning & command agent gets the system states and its component states from the diagnostic agent and then give commands to its components according to this information and the command from the higher level. The 16 service loads are relatively independent. They do not have direction relationships for each other. However, they share the same resource. Since the resource is limited, the service loads will compete for each other to get the resources. How to decide resource distribution for each service loads? In this paper, the system will evaluate the importance of each service load by two factors: the environment situation and the service load state.

$$P_{Si} = f_{Si}(S_s)$$

$$P_{Ei} = f_{Ei}(S_E)$$

$$P_i = W_{Si} P_{Si} + W_{Ei} P_{Ei}$$

Where,

P_{Si}: The ^{i}th service load priority according to all of service load states

P_{Ei}: The ^{i}th service load priority according to environment situation

P_i: The ^{i}th service load priority according to environment situation and service load states

$S_S = (S_{S1}, S_{S2}, \cdots, S_{Sn})$: The service load states, n is the number of service loads

$S_E = (S_{E1}, S_{E2}, \cdots, S_{Em})$: The environment states, m is the number of environment variables

W_{Si}, W_{Ei}: Weighting factors for balance between the importance of the environment and the service load states.

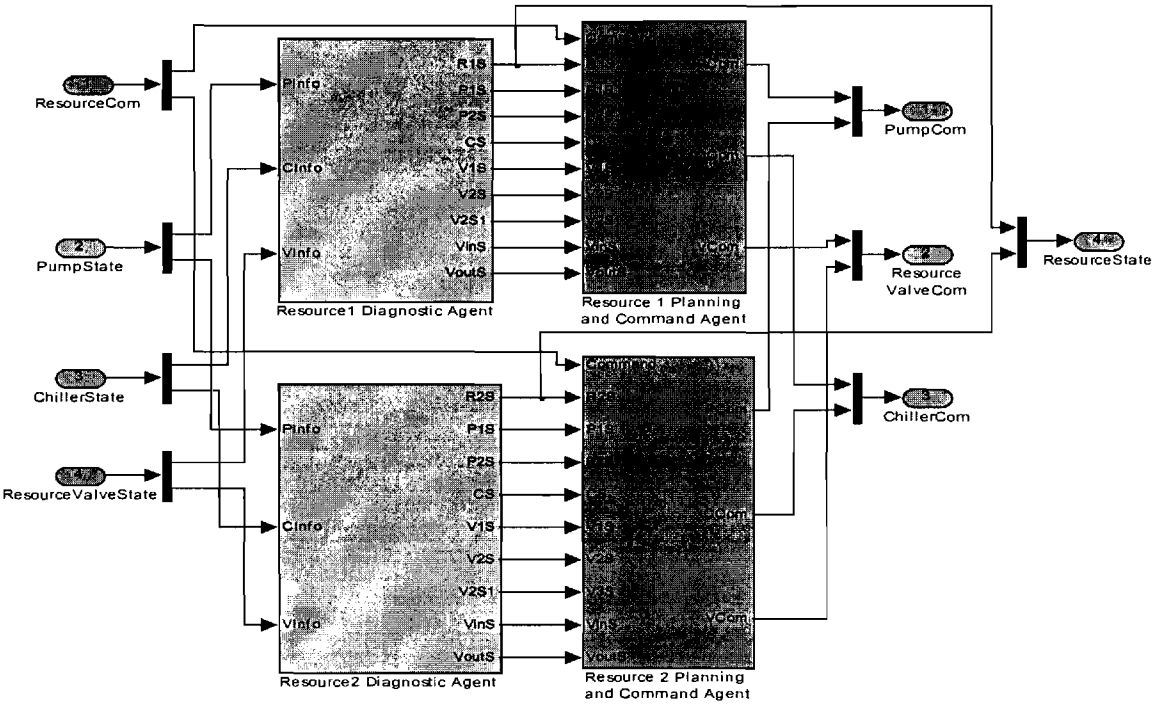


Figure 82: Integration of Two Chilled Water Resources

As mentioned before the two chilled water resources are identical in structure and one of them are redundant. When one service load can not get chilled water from one resource, it will try to get chilled water from the second resource. There is no direct relationship between these two resources. However, as

shown as in Figure 82, the diagnostic agent and planning & command agent are connected directly with each other. The diagnostic agent sends the whole system state and its components state to its planning & command agent which will use this information to give proper command to its components. Some arguments may arise like that without the diagnostic agent, the planning & command agent can get some information directly from the lower level and analyze them, and then give command to its components. However, by using different agents with different subtasks, a big and complex task will be divided into smaller subtasks, which makes the system much easier to design and understood. Actually, this is the beauty of hierarchical multi-agent based control.

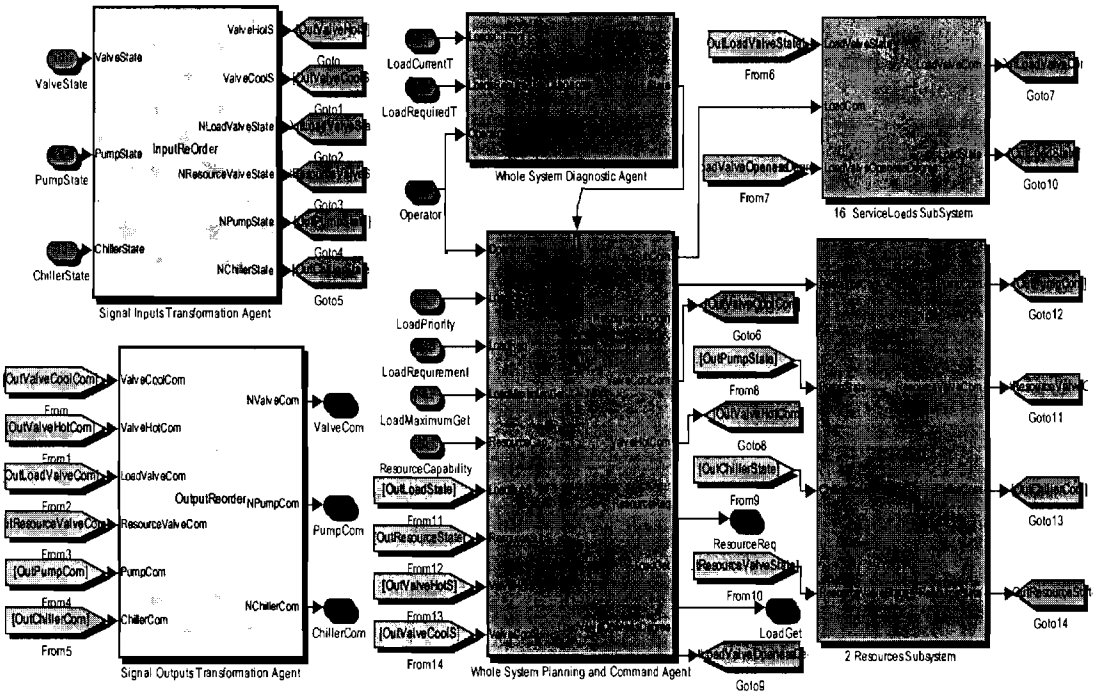


Figure 83: The Whole Agent-Based Control Model for the CW-RSAD Model

Figure 83 gives the whole picture of agent-based control model for RSAD model in MATLAB™ Simulink. To control the valve to the desired openness degree, firstly, an automatic control method based on feedback and proportional integration differential (PID) is considered. The most challenging part of this method is the delay of the system. How much the delay will be depends on the physical mechanical system and the calculation speed of the computer. Unfortunately, the delay in RSAD model is pretty big, so the vibration of using PID controller is too big to be accepted here. In this paper, a relative openness degree is used which works very well if the relationship between the valve openness degree and the flow rate though the valve is linear or close to linear. In RSAD model, all of the valves have the capability to be opened to certain degree. In order to simplify the control system and make it clearer to display the idea

the paper is addressing, just the valves connecting in the service load subsystems is controlled to some degree of openness and the other valves just have four states: open, closed, stuck open and stuck closed.

Test the Control System

After the establishment of multi-agent based control, it is time to test the control system. It is easy to predict response to certain environment of a simple agent. However, it is very difficult to evaluate the integrated effects of Multi-Agent-Based control systems theoretically – whose control behaviour emerges from the concerted activities of many agents. Extreme situations may cause the whole system crashed for some imperfect logics for the interactions between agents, so test of Agent-based control systems is as important as design the controllers. Here damage scenarios are used to test the control system. Basically, there are two methods to select the damage scenarios. The first one is using the initial conditions to control the components damage states; the second one is to control the components damage states during the simulation. Since the damage scenario is unpredicted, some statistic results should be collected. If the first method is chosen, the initial condition should be changed probabilistically. If the second method is chosen, each component failure rate should be given and their damage states during simulation should be updated. Both of these two methods have their advantages and disadvantages. The first one is easy to implemented, and it is easy to see the results with its corresponding damage scenario, but the scenarios should be chosen carefully to be consistent with the real system situation. The second one needs each component's failure rate. At each simulation step, the damage scenario is different, so it is hard to collect the data and use them to calculate the statistics. In this paper, the first method is adopted, but how to initialize the system with all of the possibilities is a big challenge. Here, assume the limited capability of resources is a constant which is reasonable. The inputs about components states have limited discrete values. There are 72 valves with four states each, two chillers and four pumps with 3 states each. If Full Factorial Design of Experiments is used, there will be $72^4 \times 6^3$ damage scenarios, which is too big to be run. Random inputs with some distributions can be used to check the possibility of success of the control system.

6.2.3.2 Resource Allocation

The IRIS framework provides a concept that integrates different ship systems to monitor and assess the ship state and then reacts to the current state by reconfiguring the ship to a new state which can best handle the situation at hand. Obviously, the ultimate objective of IRIS concept is to enable the ship to make autonomous decisions for determining the best action in each state to effectively perform the desired mission. To accomplish this objective, the problem can be modeled as a multi-agent Markov decision process and an optimal policy can be obtained to identify the best course of action. A resource allocation advisor, described in section 5.4.2, was proposed to make autonomous decisions for the

resource allocation process. To demonstrate the autonomous decision making and reconfiguration capabilities of the advisor, a resource allocation problem for the Chilled Water Reduced Scale Advanced Demonstrator (CW-RSAD) is chosen as a proof of concept.

Resource Allocation for CW-RSAD Model

Step 1: Identify state and action space for each agent

Agents

In the RSAD model, the electrical architecture was developed to match the 16 service loads. Each service load is considered as an agent and assumed to operate independently. Each service load represents a physical system, and the mapping between the service load and the physical systems are identified and listed in Table 4.

Table 4: Physical System of RSAD Model

Agent	Notation	Service Load	Modeled System
1	SVC01	AN/SLQ 32 Heat Exchanger	AN/SPY-1 Radar and Sonar System
2	SVC02	Aft Stbd Array Rm	Aft Starboard Array Room
3	SVC03	Director Eqpt Rm 1	Director System 1
4	SVC05	Aft Port Array Rm	Aft Port Array Room
5	SVC06	Fwd IC/Gyro	Forward IC/Gyro System
6	SVC08	Director Eqpt Rm 2	Director System 2
7	SVC10	Fwd Stbd Array Rm	Forward Starboard Array Room
8	SVC11	5"54 Gun Elex	Gun Weapon System
9	SVC12	HVAC CIC No.1	Combat Information Center 1
10	SVC13	HVAC CIC No.2	Combat Information Center 2
11	SVC14	HVAC CIWS wrkshp No. 1	Close-In Weapon System 1
12	SVC15	Fwd Port Array Rm	Forward Port Array Room
13	SVC16	HVAC Crew Living Space No. 2	Crew Living Space 2
14	SVC22S	C&D Heat Exchanger	C&D WTR CLR
15	SVC22P	C&D Heat Exchanger	C&D WTR CLR
16	SVC23	HVAC Crew/CPO Galley	Crew/CPO Galley Space

For simplicity and without loss of generality, the state space, action space, transition probability matrix and immediate rewards of all agents are assumed to be the same.

State Space

The states of each agent are described by the combination of two state variables, one representing the status of the agent and the other representing the priority assessment of the agent. The possible states of the agent are listed in Table 5.

The first state variable is used to describe the state of the agent itself. This state variable has three values: overheated, working properly and off. When the agent's temperature is higher than the threshold, it is considered "overheated". An agent is defined as "working properly" if it is working and its temperature is below the threshold. "Off" is a state that transits from a previous state. For example, if an agent is overheated, it may be turned off to prevent from being damaged, or if an agent is working properly but has low priority, it may be turned off to save the resources. In these cases, the agent state becomes "off".

Table 5: State Space of Agent

State *i*	Description		
1	Overheated	&	High Priority
2	Overheated	&	Mid Priority
3	Overheated	&	Low Priority
4	Working Properly	&	High Priority
5	Working Properly	&	Mid Priority
6	Working Properly	&	Low Priority
7	Off	&	High Priority
8	Off	&	Mid Priority
9	Off	&	Low Priority

The other state variable is defined as priority which is a measure of emergency of an agent. The priority is assessed based on the states of mission being performed and the operational environment and agent status.

The mission being performed has a main contribution to the priority since the agents' priorities vary significantly with the mission. Different mission requires different emphasis on certain functions, therefore, the agents which provide the required functions will have high priorities. For example, in a battle mission, in order to successfully accomplish the mission, weapon and radar systems should maintain proper functionalities and thus they have high priorities.

The operational environment, representing the surroundings of ship system, also affects the priorities of the agents. Since the same mission may be performed in different environments, the priorities of the

agents may change with the environment. Under a cruise mission, for example, the propulsion system often has the highest priority if there is no enemy around. However, when the ship is in a hostile environment, the weapons system may have a higher priority than the propulsion system.

It is clear that the priority of an agent depends on its own status. For example, if an agent is damaged, its priority is certainly low (i.e. it is not going to be used and no resource will be provided to it).

Therefore, the overall priorities of the agents are determined by the combination of mission, environment and status, given by Equation (12).

$$pr = \sum_{i=1}^{3} w_i * pr_i \qquad (12)$$

where $pr_i = (pr_{i,1}, pr_{i,2}, \cdots, pr_{i,16}), i = 1, 2, 3$ is the priority vector contributed by mission, environment and status respectively. w_i, $i = 1, 2, 3$ is the corresponding relative importance of the three contributors.

State variables can be obtained from the console, sensors or other agents which are able to directly provide the variables or supply the information that can be used to derive the values of the variables. After the state variables are obtained, the resource allocation model will formulate a multi-agent Markov decision process and then find an optimal policy to allocate the resources to the service loads.

Action Space

Three actions can be taken for each agent depending on its state. The actions are: supply agent the required cooling fluid, turn the agent off, turn the agent on and supply the required cooling fluid, as listed in Table 6.

Table 6: Action space

Action a_i	Description
a_1	Supply agent the required cooling fluid
a_2	Turn the agent off
a_3	Turn the agent on and supply the required cooling fluid

As stated in Table 5, each agent may be in one of 9 states at a given time. Notice that not all actions can be performed in each state since some actions are not appropriate to be taken in certain states. Action a_1 indicates that the required cooling fluid (chilled water) will be supplied to an agent, thus, this action can be performed in all states. Action a_2 can be taken in all the states except for the states that the agent's

state is already "off" (i.e. states 7, 8, 9) while a_3 can only be performed in such states. Table 7 lists the action spaces A_i for each state.

Table 7: Action Space for Each State

State i	Action Space A_i
1	$\{a_1, a_2\}$
2	$\{a_1, a_2\}$
3	$\{a_1, a_2\}$
4	$\{a_1, a_2\}$
5	$\{a_1, a_2\}$
6	$\{a_1, a_2\}$
7	$\{a_1, a_3\}$
8	$\{a_1, a_3\}$
9	$\{a_1, a_3\}$

Step 2: Estimate transition matrix and define immediate rewards

The transition probability matrix $P = [p_{iaj}]$ represents the probabilities of changing to state j if action a is executed in state i. It is clear that P is a |S|×|A|×|S| matrix. The immediate return $R = [r_{ia}]$ defines the expected immediate reward by executing action a in state i. The transition matrix and expected immediate rewards of the three actions for RSAD model are given by Table 8, Table 9 and Table 10, respectively.

Since the goal of the system operation is to work on the best course of action to gain the maximum desirability of its potential effect, the best course of action needs to be identified first. The optimal policy defines what action is the best to be taken in a system state, thus by following the optimal policy one can obtain the best course of action. The action taken in a state is considered as the "best" action because its potential effect is expected to best achieve the objective of the operation, that is, effectiveness. Therefore, the total rewards obtained by executing the best course of action represent the system's effectiveness. In other words, it can be stated that reward earned by the execution of an action for a state-action pair represents its potential effectiveness.

Table 8: Transition Probability Matrix and Rewards for Action a_1

State i	p_{i11}	p_{i12}	p_{i13}	p_{i14}	p_{i15}	p_{i16}	p_{i17}	p_{i18}	p_{i19}	r_{i1}
1	0.1	0.07	0.03	0.55	0.15	0.1	0	0	0	20
2	0.08	0.1	0.02	0.1	0.5	0.2	0	0	0	10
3	0.01	0.01	0.08	0.05	0.1	0.75	0	0	0	-5
4	0.05	0.03	0.02	0.6	0.2	0.1	0	0	0	15
5	0.02	0.05	0.03	0.03	0.7	0.17	0	0	0	10
6	0.01	0.04	0.05	0.1	0.15	0.65	0	0	0	5
7	0	0	0	0	0	0	1	0	0	-5
8	0	0	0	0	0	0	0	1	0	-2
9	0	0	0	0	0	0	0	0	1	3

Table 9: Transition Probability Matrix and Rewards for Action a_2

State i	p_{i21}	p_{i22}	p_{i23}	p_{i24}	p_{i25}	p_{i26}	p_{i27}	p_{i28}	p_{i29}	r_{i2}
1	0	0	0	0	0	0	0.7	0.2	0.1	-10
2	0	0	0	0	0	0	0.2	0.7	0.1	-5
3	0	0	0	0	0	0	0.1	0.2	0.7	5
4	0	0	0	0	0	0	0.8	0.15	0.05	-20
5	0	0	0	0	0	0	0.6	0.3	0.1	-10
6	0	0	0	0	0	0	0.4	0.4	0.2	-5
7	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0

Table 10: Transition Probability Matrix and Rewards for Action a_3

State i	p_{i31}	p_{i32}	p_{i33}	p_{i34}	p_{i35}	p_{i36}	p_{i37}	p_{i38}	p_{i39}	r_{i3}
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0.7	0.2	0.1	0	0	0	15
8	0	0	0	0.1	0.6	0.3	0	0	0	10
9	0	0	0	0.05	0.15	0.8	0	0	0	-5

Step 3: Available Resource and Required Resource

In this example, the resource that needs to be allocated is chilled water which is a recyclable resource. It is known that RSAD has the capacity to deliver 40 gpm chilled water, therefore, this is the cooling resource available for all the 16 service loads.

The resource required by each agent depends on their current state and action executed in this state. If action a_1 is performed in a state, the resource required by that state will be supplied to the agent. Typically, the resource required by the agent in the overheated state is greater than in the state of working properly, and the required resource is zero if an agent is in off state. If action a_2 is performed in any state, the resource required is zero since the agent is turned off and will not consume any resource. Action a_3 can only be taken in state 7, 8, and 9, and if it is executed the resource required by the agents in these states should equal the required resource that ensures they work properly. Table 11 lists the resource required to execute different actions in each state. The element g_{ia} $(i=1,2,3,\cdots,9; a=1,2,3)$ in the table defines the resource consumed by taking action a in state i. Without loss of generality, the 16 agents are assumed to have the same resource consumption for each state-action pair. From this table, one can see that when all agents work properly and are supplied the required cooling resources (i.e. the agents are in state 4, 5 or 6, and action a_1 is taken), the total required resource equals the total available resource.

Table 11: Resource Required by Each State-Action Pair

State i	g_{i1}	g_{i2}	g_{i3}
1	3	0	0
2	3	0	0
3	3	0	0
4	2.5	0	0
5	2.5	0	0
6	2.5	0	0
7	0	0	2.5
8	0	0	2.5
9	0	0	2.5

Step 4: Find optimal policy

In RSAD model, the only resource needs to be allocated is cooling fluid – chilled water which is a recyclable resource that can be reused by being chilled by the chiller of chilled water system. Therefore Equation (8) can be reduced to Equation (13) which can be solved by utilizing the linear programming technique.

$$
\begin{aligned}
&\max \sum_m \sum_i \sum_a x_{ia}^m r_{ia}^m \\
&s.t \quad \sum_i \sum_a (\delta_{ij} - p_{iaj}^m) x_{ia}^m = \alpha_j^m \\
&\qquad \sum_m \sum_i \sum_a x_{ia}^m (g_{iaw}^m - \lambda_m \hat{g}_w) \le 0 \\
&\qquad x_{ia}^m \ge 0
\end{aligned}
\tag{13}
$$

The initial condition α_j^m ($j = 1,2,\cdots,9$; $m = 1,2,\cdots,16$) is given as 1 indicating that the number of the times that the agent m starts in each state j. At this point, all the necessary information required to compute the optimal policy is obtained. By using the linear programming technique, the optimal policy of the agent is calculated and shown in Table 12. Notice that since the transition probability matrix, immediate reward and resource required by all agents are assumed the same, the optimal policies for the agents are also the same.

Table 12: Optimal Policy

State i	π_{i1}	π_{i2}	π_{i3}
1	0.96	0.04	0
2	0.69	0.31	0
3	0.58	0.42	0
4	0.80	0.20	0
5	0.96	0.04	0
6	0.94	0.06	0
7	0.31	0	0.69
8	0.44	0	0.56
9	0.56	0	0.44

The optimal policy shown in Table 12 is a randomized policy. The element π_{ia} in this table represents the probability of taking action a in state i. When the policy is executed, an action will be chosen based on the probability distribution over the state space which is defined by the optimal policy. The optimal policy presented in Table 12 provides some insights about the best action to be taken in each state. It can be seen that if an agent is in any state of 1 to 6 the probability of being supplied the required cooling resource is much higher than the probability of turning the agent off. In addition, if an agent is in off state (i.e. state 7, 8 or 9) the execution of the optimal policy will tend to turn the agent on and

supplying it the required resource except it is in state 9. It can be explained as state 9 has low priority so keeping it off can save some resource that could be used by the high priority states.

Step 5: Resource allocation Process

In the system operation, at time t, also considered as a decision epoch, an agent is in the state i. An action a is chosen from a set of allowable actions and then executed with the objective of maximizing the expected total reward. The proper action to take can be identified by following the optimal policy as shown in Table 12. That is, in a certain state which action is selected to be executed is determined by its probability over this state. For example, if an agent is in state 1, action a_1 has a probability of 96% to be executed while action a_2 has a probability of 4% to be taken. After the action is performed in the state, the agent will change to a state with a probability defined by Table 8, Table 9 or Table 10 based on the selected state-action pair, and at the same time the agent receives a reward. In this example, if action a_2 is selected, the agent will transit to state 7, 8 or 9 with probability of 70%, 20%, 10% respectively, and meanwhile receive a reward of -10. In addition, the execution of action a_2 in state 1 consumes 0 unit of resource, which can be found from Table 11. At next decision epoch, the agent will go through the same process and then move to another new state. This stochastic process is illustrated in Figure 84. Notice that at each decision epoch, all the 16 agents need to take one action based on the optimal policy, and the executions of all the actions should not overuse the total available resource.

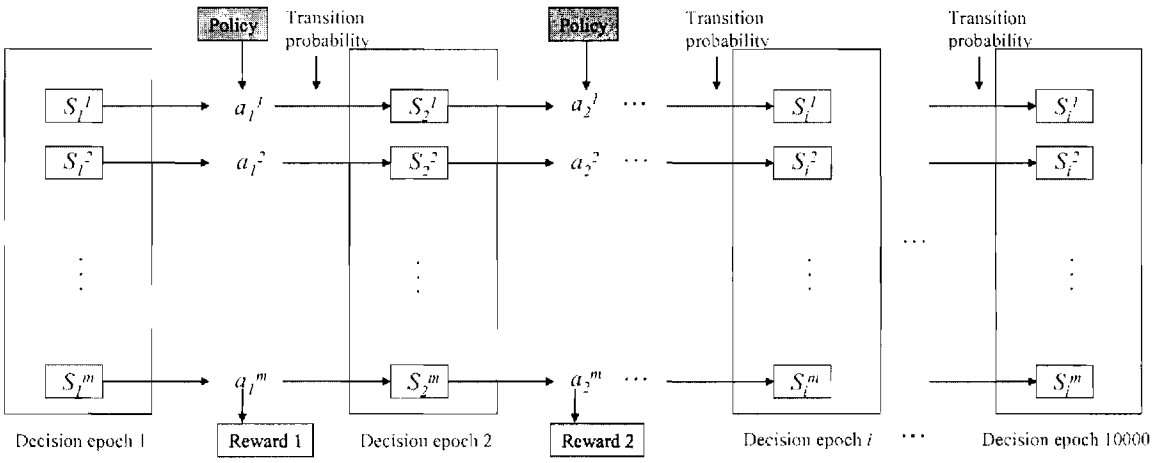


Figure 84: Action Selection and Resource Allocation Process

Simulation Study

The objective of the simulation study is to investigate the effects of the course of action defined by the optimal policy and gain insights into its performance. Since the optimal policy is the solution to the

constrained multi-agent MDP problem, eventually, the constrained multi-agent MDP formulation will be examined. To test the resource allocation model, instead of using the integrated simulation environment, a stand-alone MATLAB™ program is used to perform the simulation. In a simulation, at each decision epoch the optimal policy shown in Table 12 determines an action to take based upon the probabilities of the actions over the state. After an action is taken in the current state, the system earns a reward and then transits to a new state depending on the transition probabilities defined in Table 8, Table 9 and Table 10. This process is repeated at each decision epoch until it reaches the maximum number of the decision epoch. Thus, the optimal policy can be investigated using this simulation program without needing to invoke any other model of the integrated simulation environment.

To explore the performance of the optimal policy, four other policies, as given by Table 13, are constructed to compare with it. The policy given by Table 13 (a) is a deterministic policy which always chooses the action that will maximize the immediate reward in each state. The other three randomized policies are arbitrary, valid policy.

Table 13: Four Policies

(a) Maximum Immediate Reward Policy

State i	π_{i1}	π_{i2}	π_{i3}
1	1	0	0
2	1	0	0
3	0	1	0
4	1	0	0
5	1	0	0
6	1	0	0
7	0	0	1
8	0	0	1
9	1	0	0

(b) Arbitrary Policy 1

State i	π_{i1}	π_{i2}	π_{i3}
1	0.8	0.2	0
2	0.7	0.3	0
3	0.4	0.6	0
4	0.9	0.1	0
5	0.6	0.4	0
6	0.4	0.6	0
7	0.4	0	0.6
8	0.5	0	0.5
9	0.8	0	0.2

(c) Arbitrary Policy 2

State i	π_{i1}	π_{i2}	π_{i3}
1	0.4	0.6	0
2	0.1	0.9	0
3	0.7	0.3	0
4	0.5	0.5	0
5	0.3	0.7	0
6	0.8	0.2	0
7	0.5	0	0.5
8	0.1	0	0.9
9	0.4	0	0.6

(d) Arbitrary Policy 3

State i	π_{i1}	π_{i2}	π_{i3}
1	0.9	0.1	0
2	0.8	0.2	0
3	0.6	0.4	0
4	0.2	0.8	0
5	0.3	0.7	0
6	0.4	0.6	0
7	0.7	0	0.3
8	0.9	0	0.1
9	0.2	0	0.8

Policy Comparison

Average Reward for One Simulation

The comparison starts by running a simulation for each of the policies. In the simulation, the 16 agents begin with an initial state and go through 10000 decision epochs by following the policies. Since the 16 agents are assumed to operate independently, their initial states are also independent. The initial states of the 16 agents compose the initial state of the simulation which is shown in Table 14.

Table 14: Starting States for Policy Simulation

Agent	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
State	3	8	7	8	3	6	3	3	4	7	9	3	4	7	8	4

In the simulation, a policy is utilized to make the decision and select the action in each state. The operation of the system starts from the initial state $S_0 = \{i_0^1, i_0^2, \cdots, i_0^{16}\}$ (where $i_0^m = \{1,2,\cdots,9\}, m = 1,2,\cdots,16$) at decision epoch t_0. At this epoch, decisions need to be made upon choosing one action a_0^m (where $a_0^m = \{1,2,3\}$, $m = 1,2,\cdots,16$) for each agent in its initial state based on the probability $\pi_{ia,0}^m$ (where $i = 1,2,\cdots,9; a = 1,2,3;\ m = 1,2,\cdots,16$) defined by the policy. The effect of the action will result in a transition to a new state s_1^m (where $m = 1,2,\cdots,16$) and a reward $r_{ia,0}^m$ (where $i = 1,2,\cdots,9;\ a = 1,2,3;\ m = 1,2,\cdots,16$) is earned by the agent. The transition is manipulated by the transition probability matrix shown by Table 8, Table 9, or Table 10 depending on what the state action pair is. Consequently, the system enters to the next decision epoch t_1 and the same process at decision epoch t_0 will be repeated. The system then moves to the next decision epoch until the maximum number of the decision epoch (i.e. 10000) is reached. This process can be clearly viewed in Figure 84.

Table 15: Statistic Results of Average Reward for Five Policies

Policy	μ	σ
Optimal	98.0275	30.6818
Max Immediate Reward	48.0439	2.8519
Arbitrary 1	60.5912	36.6398
Arbitrary 2	17.5475	45.3764
Arbitrary 3	-27.8321	34.6259

The rewards gained by the agents at each decision epoch are calculated for each policy. Figure 85 shows the reward trajectories for the five policies in a 10000 decision epoch simulation. The statistic results of average reward for the five policies are listed in Table 15.

From Table 15 and Figure 85, one can see that maximum effectiveness (maximum average reward) is obtained when the system operates by following the optimal policy. This indicates that the best course of action is executed during the system operation process, and the optimal policy does have better performance than any other policy. The maximum immediate reward does not perform well. This implies that the decisions must not be made myopically, but must anticipate the opportunities and rewards associated with future system states. Different policies have different performance, and poor policy may lead to cost (negative reward) to the system and, in turn, makes the resource allocation ineffective. Notice that after around 100 decision epochs, the reward keeps as a constant for maximum immediate policy. This is due to the fact that once a state transits to state 9 it will stay in this state forever because action a_1, which generates the maximum immediate reward for state 9, will be taken in this state and the probability of changing back to state 9 is 1.

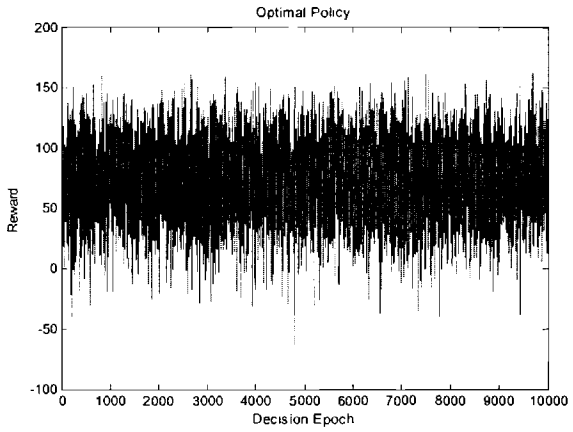
Total Rewards for 50 Simulations

To further investigate the performance of the policies, 50 simulations are performed for each policy. In each simulation, the 16 agents start from an initial state and go through 10000 decision epochs by following one of the policies. The initial state of each simulation for all the policies are the same so that the policies can be compared based on the same basis. The total reward in one simulation can be obtained by summing the reward at each decision epoch. The simulation runs 50 times for each policy and the total rewards of each run is computed. (a) to (e) of Figure 86 show the total rewards for the five policies in 50 simulations. Figure 86 (f) depicts the total rewards obtained by following each policy in the operation for 50 simulations. The statistic results of the total rewards for the five policies are listed in Table 16.

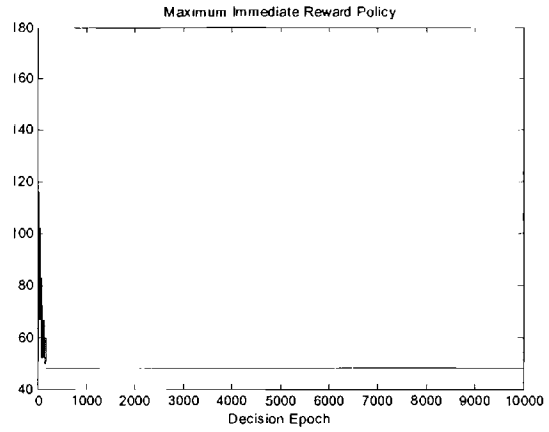
Table **16**: Statistic Results of Total Rewards for Five Policies

	Optimal	Max Imme Rwd	Arbitrary 1	Arbitrary 2	Arbitrary 3
μ	9.84E+05	4.81E+05	6.05E+05	1.74E+05	-2.77E+05
σ	3.13E+03	358.623	3.68E+03	3.12E+03	2.28E+03

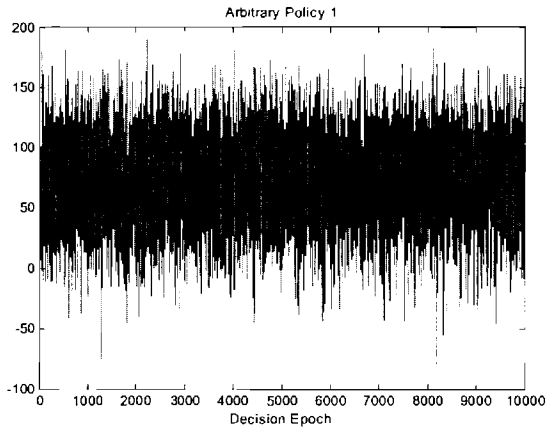
It can be seen from Figure 86 and Table 16 that the total rewards (effectiveness) vary with the simulations starting from different points. For different starting points, maximum total rewards are always obtained if the system operates under the optimal policy.



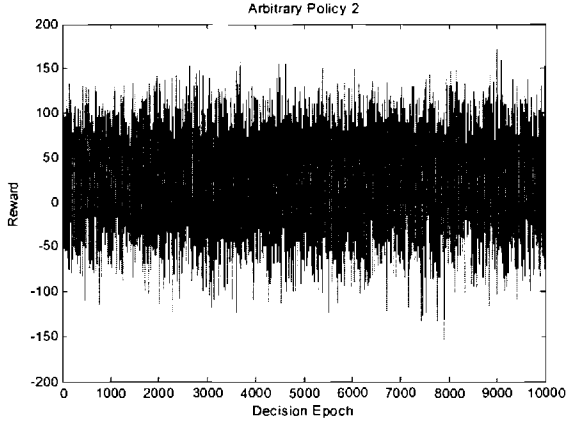
(a) Optimal Policy



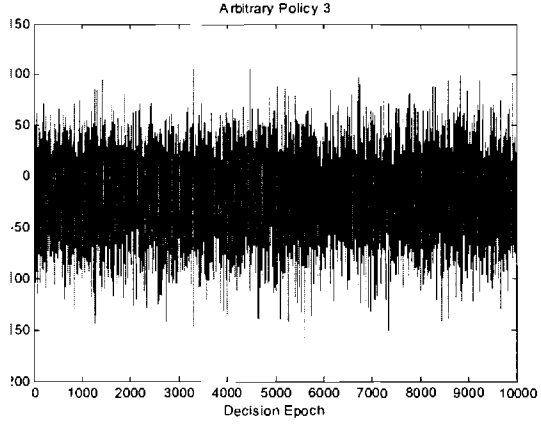
(b) Maximum Immediate Reward Policy



(c) Arbitrary Policy 1



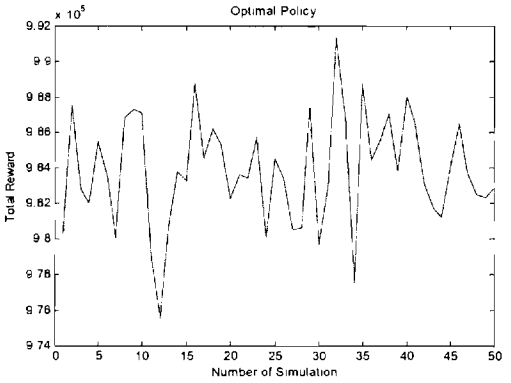
(d) Arbitrary Policy 2



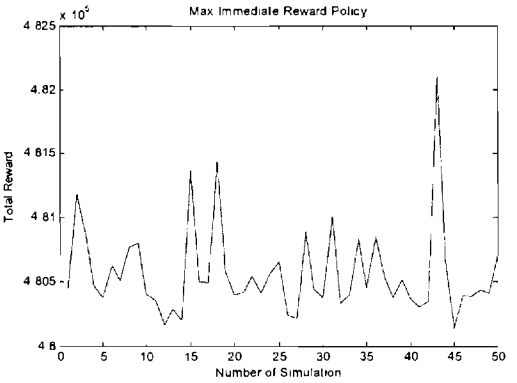
(e) Arbitrary Policy 3

Figure 85: Reward Trajectory for Five Policies

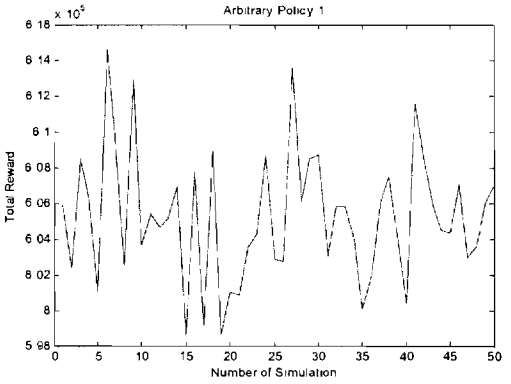
This indicates that the optimal policy given by the MDP formulation does offer the best performance for system operation. Again, the maximum immediate reward policy is not a good policy, which implies that the actions with maximum immediate reward may produce side-effect on the future system states.



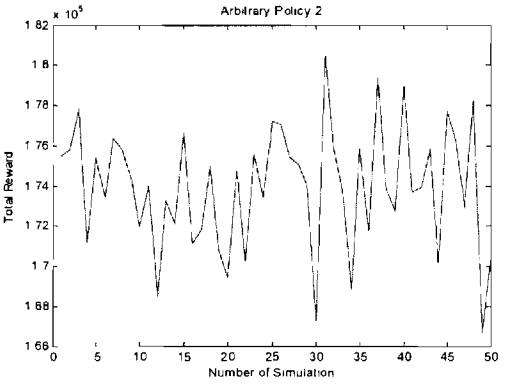
(a) Optimal Policy



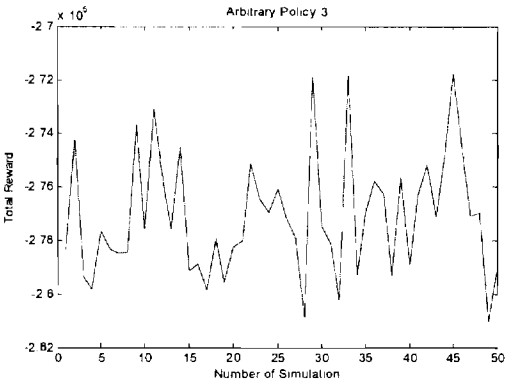
(b) Maximum Immediate Reward Policy



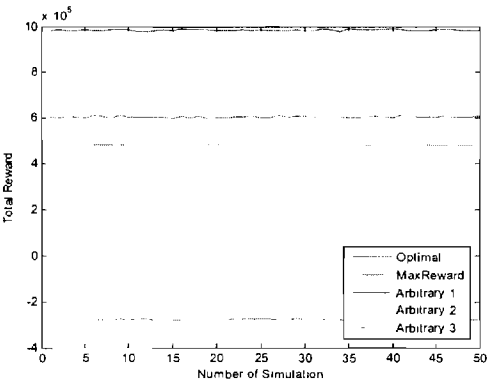
(c) Arbitrary Policy 1



(d) Arbitrary Policy 2



(e) Arbitrary Policy 3



(f) Total Reward Comparison

Figure 86: Total Rewards for Five Policies

6.2.4 Human Machine Interface

6.2.4.1 Web-based Application

Web-based applications (webapps) are applications that run in a web browser over a network, either an intranet or an internet. The concept gained popularity because most computers are equipped with web browsers. This application model stems from the network computer (NC) model that tends to resurface periodically over time. Originally, it was not feasible to place a computer at every desk. Terminals were used to provide access for employees that had a need for a computer. The terminals relied on serial communications to transfer information between the main computer system and the client terminal. With the introduction of desktop computers, it became feasible for users to run applications locally using local resources (processor, ram, etc). The NC infrastructure that existed before desktop computers still represented a considerable investment. As a result, desktop computers did not immediately replace the old NC architecture. Instead, new applications allowed the desktop computers to emulate their terminal client predecessors. Many businesses still use this model. Today the modern approach has been to use powerful desktop computer that can utilize services available over the network. Rather then running an application on a server, the applications are running on the desktop computers that perhaps use services such as a database or file system available on a server.

The webapp model is a revisitation of the old NC idea except for a slight fundamental difference. Although the web server provides the application to the client, the application is running on the client machine (using local resource). When a web browser is pointed to any specific web server, the browser begins the dialog by making a request. In response, the web server sends a stream of text back to the web browser. Generally, this text is encoded using the HTML markup language. Webapps utilize the HTML encoded text to transfer a program that can be executed in the web browser. What sets a webapp apart from a classic web page is that a webapp has the ability to accept and response programmatically to user inputs, instead of simply being a collection of static web pages. Some advantages to the webapp model include:

1. A centralized code base for applications. The webapp model does away with the scenario of a group of users using different versions of applications (with the exception of the browser...). Each time a webapp is loaded newest code base is loaded from the web server.
2. The applications are accessible from any location with a network connection.
3. Webapps have the potential to capitalize on the strengths of both server based and client based webapp architectures.

6.2.4.2 Ajax

The Ajax model is a hybrid between the server side application and the client side application. The concept is to move the application burden from the server to the client, thus increasing the application's responsiveness by eliminating the network latency between user input and the server's response. The application then utilizes an asynchronous HTTP request method in the background in order to obtain the data required from the server. Google is one of the first to use this approach with their webapps. The Google maps webapp provides a graphical user interface (GUI) which remains responsive to the user while map imagery is being loading in the background. The Ajax approach eliminates (or reduces) the need for a browser refresh each time something changes. Each time a browser refresh is performed a request is sent to the server to send the entire webapp. When the client receives the webapp time is required to load and run the webapp in the browser. In short, the Ajax approach allows the server to spend most of its time serving data and allows the client to run the application using the data.

6.2.4.3 Open Source Software

There are some differences in opinion to the actual meaning of open source. Generally, it means that the source code is openly available. That is not to be confused with free, meaning software without cost; however, much of the open source software available is free to use without cost. Open source software projects come in various shapes and sizes. Some open source projects are very mature with active communities working to further enhance or update the software. Some projects do an excellent job of documenting their work; however, these conditions are not always the case. It is important to investigate the overall usefulness of a project's code before it is used.

6.2.4.4 Object Oriented Programming

Object oriented programming (OOP) is a computer programming paradigm. The concept is based on the idea that a program could be thought of as a collection of objects as opposed to a set of instructions. Objects in some ways are containers that hold methods (sets of instructions specific to that object) and data. The advantage of OOP is clear when having multiple instances of the same class of things in a program. For example a program with lots of buttons might have a class defining, in general, what a button is, after which each button in the program will become represented with its own object that specifies the uniqueness of each button.

6.2.4.5 Scalable Vector Graphics

There are a number of ways a graphic can be represented. The most familiar method to represent a graphic is to use a raster image or bitmap. A bitmap is a data file defining a grid of pixels with each pixel's color being defined with a red, green, and blue value (RGB space). A scalable vector graphic

(SVG) is an image defined by a markup language called SVG. The underlying principle is that any image can be represented with a collection of elementary shapes. There are several advantages to using SVG over raster images. First, SVG is scalable without degradation in image quality. For a webapp, this means that even though the end user's display is unknown to the developer the graphics can still be rendered as if designed especially for that display. This characteristic is especially useful for large displays where a raster image would receive a considerable amount of degradation. Furthermore, because a SVG file does not store information about every pixel, but rather it stores the geometry of the image, SVG files tend to be smaller and thus better suited for a network environment.

6.2.4.6 Implementation

Web Server

The Apache HTTP web server is the leading web server in use today. It also happens to be a free software package as part of an open source development project (Apache HTTP Server Project). Apache is a cross-platform product, meaning it can be used on UNIX-like systems (including apple computers), Microsoft Windows, or Novell based machines. Apache is considered best in class and is available at zero acquisition cost. Thus, almost by default the Apache server is the ideal choice.

Web Browser

Not all web browsers are created equal. In fact, the most difficult aspect to web development is writing code that is equally compatible with the various web browsers in use. The most popular web browser is Microsoft's Internet Explorer (IE). However, IE is not fully cross-platform and it is of interest to not reduce the usefulness of the webapp to any group of users unnecessarily. A strong alternative is Mozilla's Firefox web browser. Firefox is a cross-platform software product, and happens to be a free software package from the Mozilla Corporation. In addition to the cost advantage at this current time Firefox is a step a head of IE in both performance (specifically SVG rendering performance) and in features. A most persuasive reason to choose Firefox lies in one important key feature. That feature being an advanced JavaScript debugger. Since the web browser is being used as both a development tool and a user client package Firefox is the clear choice.

Server Side Programming

For the server side programming language several options exist. This choice however is more of a matter of preference. Several different approaches could be used to manage the relatively simple function of this component of the HMI framework. This example uses an interpreted programming language called Hypertext Preprocessor (PHP). PHP does have some big advantages in the event that a more sophisticated approach is desired on the server side of the framework. PHP is very well documented by the actual

developers, PHP is also very popular, thus example codes are readily available, and PHP is the language of choice in the open source communities, thus much help is available.

This version of the HMI framework requires two functions of the server side code. First, it must be able to accept data and store it. Second, it must be able to retrieve the correct data and send it in response to a request. There are two primary techniques to send data to a server using HTTP. One technique is to encode the data into the Uniform Resource Locator (URL). The other technique is to send the data using the POST method. Both would work equally well, except using the URL is easier to debug. Eventually the server will be using MySQL to store and manage the data on the server. Since this is the first proof of concept, this version will use data files to store and manage data. Figure 87 shows the example code used in this proof of concept. First, the code validates the input embedded in the URL. Next, it writes the data from the URL to a file and then reads another file as the return data. As an easy debugging strategy, simply type an URL in to any web browser to test the PHP functionality. In the future, this code will also include the ability to read from, write to, and create database entries.

```
<?php
/
T h i s   f i l e   i s   t h e   s e r v e r   s i d e   p h p   c o d e   t h a t   w i l l   w r i t e   t h e   d a t a   f r o m   t h e   U R L   t o   a   f i l e   a n d   t h e n   r e a d   a n o t h e r   f i l e
  a s   t h e   r e t u r n   d a t a .   T h i s   i s   t h e   s i m p l e s t   p o s s i b l e   p r o o f   o f   c o n c e p t .
       T h e   n e x t   v e r s i o n   w i l l   u s e   M y S Q L   t o   s t o r e   a n d   m a n a g e   t h e   d a t a .
  T h e   f i n a l   v e r s i o n   w i l l   a l s o   i n c l u d e   t h e   a b i l i t y   t o   c r e a t e   d a t a b a s e   e n t r i e s .
  T h e   f i n a l   v e r s i o n   w i l l   a l s o   v a l i d a t e   t h e   i n p u t   d a t a .
/
// if(isset($_REQUEST['fileOut'])){
.       $fileOutName = $_REQUEST['fileOut'];
} else {
.       echo "There was no output file in URL <br>";
}
// if(isset($_REQUEST['fileIn'])){
.       $fileInName = $_REQUEST['fileIn'];
} else {
.       echo "There was no input file in URL <br>";
}
// if(isset($_REQUEST['dataOut'])) {
.       $dataOut = $_REQUEST['dataOut'];
.       // if(is_writable($fileOutName)) {
.       .       if(!$fileHandle = fopen($fileOutName,"wt")) {
.       .       .       echo "cannot open file \"$fileOutName\"<br>";
.       .       .       exit;
.       .       }
.       .       // if(fwrite($fileHandle,$dataOut) === FALSE) {
.       .       .       echo "cannot write to file \"$fileOutName\"<br>";
.       .       .       exit;
.       .       }
.       .       // echo "Success, data written to file \"$fileOutName\"<br>";
.       .       echo file_get_contents($fileInName);
.       .       fclose();
.       } else {
.       .       echo "The file \"$fileOutName\" is not writable<br>";
.       }
}       else {
.       echo "There was no data in URL";
}

?>
```

Figure 87: Server Side PHP Code

Client Side Programming

A similar statement might be said for the client side programming language as was said for the server side programming language, where the two premier options would be VBScript or JavaScript. However, since Firefox was chosen over IE the language of preference becomes JavaScript in order to avoid cross-browser incompatibilities. JavaScript is an interpreted OOP language. It has the advantage that it is already generally associated with the Ajax approach. This again amounts to a well-documented use of the language for the type of application the HMI framework is designed to produce.

This section of the project is where the bulk of the time was spent. Here it is important that the architecture of the code be well planned with room for growth. The following is a list of requirements for this portion of code:

1. The code should be well documented and the logic should be written in an easy to follow fashion.
2. The code should be modular as to make it easy to add functionality without disturbing other sections of code.
3. The code should be scalable such that effects due to the size of the simulation are minimized.
4. The code should be displayable such that it is as independent as possible from the target display.
5. The code should be robust to network conditions.
6. The code should be simple to implement for the creation of new HMI webapps.
7. The code should be able to validate inputs.
8. The code should be able to handle errors.

The first version of the working client side code can be found in the appendix. Starting from the top, the code begins with basic usage instructions. This set of instruction is meant to be a quick reference. Next, the code creates a new object called HMI. The HMI object is a container object. A tree with two primary braches can represent the HMI object.

The first branch is called *elements*. The second branch is called *dataManagers*. From this point on, the end user will be dealing with either HMI *elements* or HMI *dataManagers*. Both *elements* and *dataManagers* are objects in and of them selves. Examples of HMI *elements* are meters, controls, backgrounds, etc. Every HMI *element* must be associated with a *dataManager*. A *dataManager* handles the communications between HMI elements and the database on the server.

Simulation Plug-in

A simulation plug-in is the necessary modifications required to get data in and out of the simulation. In this example case, Model Center from Phoenix Integration is being used as the simulation environment. Since this example is running on a Microsoft Windows XP machine it has been determined that a custom

plug-in should be used to extract data from the simulation and insert data into the simulation in real time. Model Center includes the ability to access the built-in JavaScript interpreter from its script plug-in module. Although this specific use for that capability is not entirely well documented this report should certainly fill in the gaps. The plug-in uses the same protocol and communication methods that the webapps use. This is a very convenient solution for debugging purposes as well as for security concerns. The code used for the plug-in can be found in the appendix.

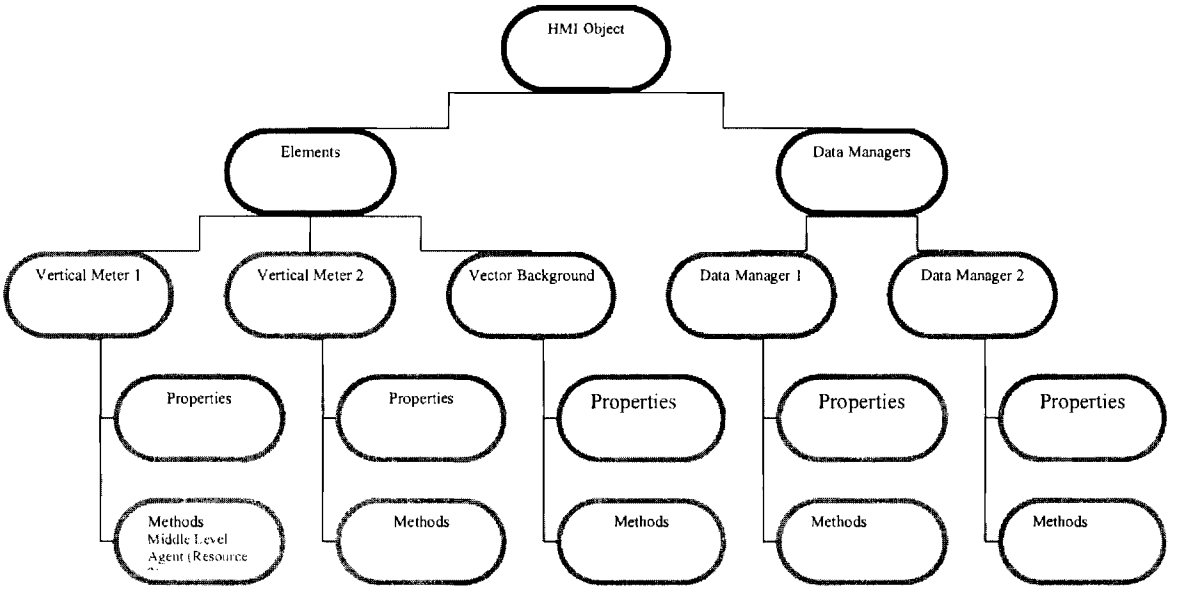


Figure 88: Example HMI Object Structure

Preliminary Results

The HMI framework is complete in concept. The example used proves that webapps built on open source technologies can be utilized for HMI applications, specifically for accounting for a human-in-the-loop contribution to a time domain simulation. The example also demonstrates how the end user can rapidly develop custom HMI webapps using either pre-made elements or custom designing new elements for interaction on the screen. Moreover, the example also illustrates the versatile nature of webapps in general. For instance, multiple webapps can act together as one HMI for more extensive HMI requirements. Webapps can be utilized without a significant additional burden to the machines running the simulation. Webapps can take advantage of the security features of internet communications. Webapps are flexible to multimedia enhancements such as sound, graphics, etc. Below is a screenshot of the example HMI running.

The image shows the successful use of scaleable vector graphics for generating graphics and diagrams. Furthermore, the figure above illustrates the use of slide bars for continuous input controls.

The highlighting accomplishment of this framework is the capability to interact with a time domain simulation in real time, more specifically to have demonstrated that this framework can be used to interact with a time domain simulation running in Model Center. The plug-in developed for use with Model Center is a simple drag and drop icon, which effectively extended Model Center's ability to send and request data from an external server.

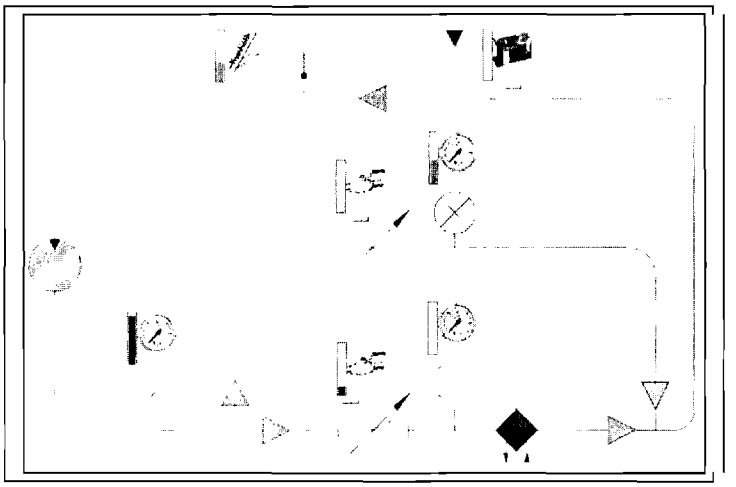


Figure 89: Screenshot example HMI

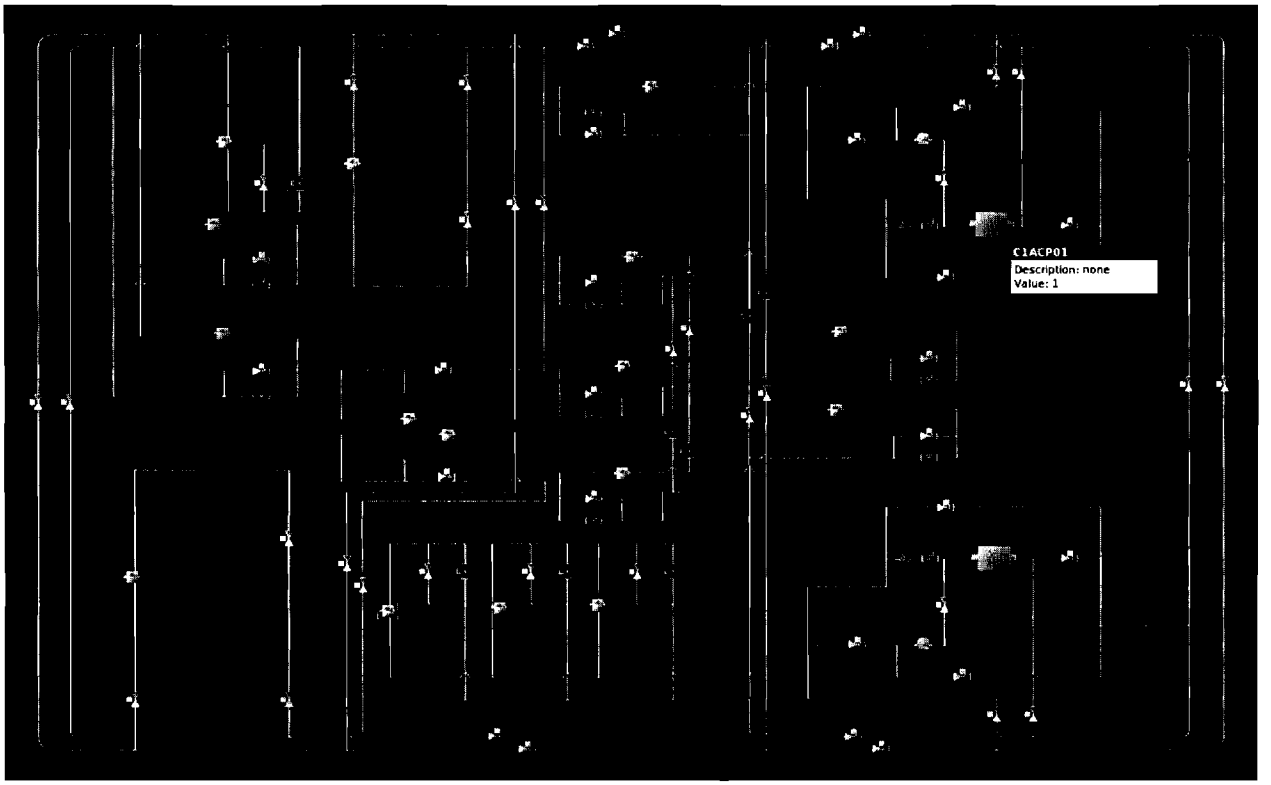


Figure 90: Screenshot of current HMI for chilled water system

7 Conclusions

The Navy is pursuing aggressive reductions in its operational cost to meet the congressional expectations for the next century. At the same time, it is seeking to have improved mission effectiveness. Operational costs, mainly driven by manning, are two thirds of the ownership cost of a surface combatant. IEP has been proposed as a solution to part of the problem the Navy faces and IRIS has been set up as a method for designing and developing processes for operating the IEP. It is important to remind the reader that IRIS is a generic framework that can be applied to any complex dynamical system.

7.1 Design

IRIS systems have been characterized as constantly iterating the three IRIS functions, sense, assess and react. The proposed methodology for IRIS systems decomposes this cycle and describes what should be modeled. An example for sensor optimization has been provided. This method requires the use of an integrated model to perform the tradeoffs required. The need for design for mission effectiveness has been established, considering the benefits it could impose for the IRIS implementation of the IEP and for applications in other engineering fields. However, this methodology would require a suitable M&S environment for its development. ASDL has made great progress in creating a modeling and simulation environment by integrating models of subsystems but better models should be used and added in the future.

7.2 Model Integration

The integration methodology that has been proposed has offered a framework that meets initial requirements and expectations. The next phase of this research initiative will be to link computational models of higher fidelity, such as IEP representative electrical and thermo fluid systems. The Multidisciplinary Simulation (MDS) method has been presented as a viable method for integrating time-domain simulations that make use of different software packages. The methodology has been demonstrated using the Phoenix Integration ModelCenter, a process integration tool. The experiments performed to validate this method have used only two models with a total of eight state variables; the integrated CW-RSAD model has hundreds of state variables demonstrating the scalability of the process, although further scaling of the model may need to be demonstrated. It has been shown that it is possible to utilize a process integration tool to link and automate the execution of simulations. Furthermore, if computational load distribution strategies (such as utilizing Centerlink, Phoenix Integration's job scheduling tool) can be used, or if analyses can be dedicated to certain workstations connected to the ModelCenter network, faster-than-real-time simulation of complex systems may be possible.

7.3 Electrical Model

Two low fidelity models were implemented in two stages in this research venue. Both the NCS and the ZELDA models were inadequate in their present form to be used in the building the low fidelity models. ZELDA was used as a starting point for the first model, and the physics sub-models from the NCS were used in the second low fidelity model. Both of the low fidelity models were designed and built using the object oriented methodology, hence a library accompanying each model was developed, allowing for future expansion of these models.

The first stage model was implemented along the same lines as ZELDA's architecture. As commonly configured on ships, this model depicts the power network as two main starboard/portside buses, both supplying zones of loads in a spatial configuration. It simplified and improved the flexibility over ZELDA's design and allowed for an arbitrary number of zones, loads and priority indices for each load. It correctly modeled the priority based power allocation algorithm but was not confined to this type of control. Nevertheless, it lacked load or component interdependency and physics based modeling.

The second stage low fidelity model used the same logic as ZELDA, but incorporated the physics-based equations from the NCS model to calculate the heat lost. This model runs in two steps. First the power allocation algorithm is activated. Electric power is assigned to components according to a comprehensive set of rules taking into consideration the connectivity of components and their priority indices. In the second step, the interdependency of loads, pumps and converters is evaluated, shedding down components and modifying the network interdependency configuration. The temperatures of components are also calculated according to the thermal model implemented. The power assignment schedule is updated with the new configuration. Similar to the first stage model, loads can be grouped in zones, although no port/starboard buses are represented.

The simulation team used the second stage low fidelity model in the modeling and simulation environment. The power model integrated seamlessly in the environment and exchanged data with the control and the fluid models. The next step is to substitute this low fidelity model with a higher fidelity one to produce more accurate results.

7.4 Fluid Model

The CW-RSAD software model running in Flowmaster2, was modified to be capable of simulating the interactions with electrical and control systems. For the CW-RSAD network to be reconfigurable, the COM-enabled components were added to all the valves in the network so that the valves receive the valve commands from agent-level controllers. Heat transfer is one of the main considerations in analyzing the performance of the total system; therefore, all the service load networks and chillers were modified to be thermal-analysis capable. These service load networks cool the various parts of the electric power system

and transfers the heat to the sea water cooling system through the chiller. Also the pumps were modified to be operated at variable rotational speeds, so that a high-level controller can control the amount of chilled water being pumped through the network. Finally, the nomenclature for all interacting components was proposed as an effort to manage a large number of inputs and outputs interconnected to various subsystems.

An automated plug-in for integrating Flowmaster2 models in the ModelCenter framework was developed to ease in the integration of future fluid networks. Not only the implementation for the integration was achieved, but this new plug-in is a way of improving the simulation performance. As a tool to improving the simulation speed and the easiness of integration, a method using surrogate modeling is being developed. This method has already been tested with success to a simple fluid network system made of a service load and a pump, and it will be further developed by testing more complex systems.

7.5 Agent-Based Control

Hierarchical multi-agent based control is a framework of modern control methods. It is implemented by decomposing a complex system into several smaller subsystems; each subsystem is treated as a relative isolated part; all of the parts work interactively by their interfaces (inputs and outputs). This procedure can be addressed in three steps: decomposition, abstraction and organization. Hierarchical multi-agent based control is a natural and effective way to handle complexity of a system in a hierarchical form. First, it divides a large and complex task into smaller portions, which makes it easier to implement many modern control methods such as neural network control, fuzzy logic control, etc. Second, it separates the controller from the physical model and the controller is designed as software which can handle computational complexity. Third, it makes the system more robust in uncertain and dynamical environments. Fourth, it makes the system more flexible and adaptable by permitting designers to upgrade the required portions only.

The chilled water system of the Arleigh Burke class ships is a very complex system which provides chilled water to electronic equipment, weapon systems, ship spaces, etc. When considering a combat scenario, the environment is extremely uncertain. It is infeasible to predict all possible damage scenarios and store the corresponding strategies in memory to align the system to its most desirable configuration. Furthermore, the chilled water system spreads over the entire ship, so it is a widely distributed system and interacts with other subsystems of the ship. Hierarchical multi-agent based control is a more efficient and organized way to deal with the uncertainty and complexity of systems like the ship chilled water system. At the same time, it provides a clear interface to the Integrated Environment Plant (IEP).

7.6 Resource Allocation

Complex systems, like an aerospace vehicle and a naval ship, mainly operate in a dynamic environment where decision making is required to be performed under uncertain conditions. This is always a challenge to human decision makers since it is difficult for human beings to manage and organize the time-dependent information and make appropriate decisions based on the probabilistic assessment of the acquired data. In order to increase mission effectiveness and ship survivability and reduce operating cost, decisions need to be made autonomously to perform the desired mission or manage emergent events. A constrained multi-agent Markov decision process was proposed and applied to the resource allocation problem, which results in the formulation of a resource allocation advisor. A resource allocation problem for a chilled water reduced scale advanced demonstrator was performed as a proof of implementation. The results show that the proposed resource allocation advisor can derive the optimal policy which manipulates the system, advises on the best course of action, and reallocates the required resource to the system to reconfigure the ship into the best mode that can best handle the situation at hand.

7.7 Human Machine Interface

The HMI framework is a flexible and powerful enabler to human-in-the loop simulation. The HMI effectively extends the usefulness of any engineering simulation package with a component object model application programming interface (COM API) or a file based interface. ModelCenter is fully supported with the HMI using a plug-in developed and tested for the chilled water system and the high level agent control system. The framework is fully network centric allowing distribution of work load both computationally and with respect to human interfaces over multiple machines. Future work includes a web application design to provide a graphical user interface for constructing new interfaces quickly and controlling server side configurations. Future work also includes implementing a more sophisticated database system. The framework is meant to be a more than a specific application it is design to provide a means to quickly actualize both human control and show a graphical visualization of simulations and or actual systems. In the end the HMI also proved to be a powerful debugging tool for simulation integration.

Its visualization capability allows both the end user and the developer inside understanding of the simulations ensuing in higher level quality decision making by end users and higher quality simulations.

8 References

[1] "Smartship: 'Technology Transfer in Action'". NAVSEA Crane, Indiana. http://smartship.navy.mil/Dave_Final_Paper.doc. October 2004.

[2] Acton, D.E. and J.R. Olds, "Computational Frameworks for Collaborative Multidisciplinary Design of Complex Systems," AIAA 98-4942, 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, St. Louis, MO, September 2-4, 1998.

[3] Aerospace Systems Design Laboratory, "Integrated Reconfigurable Intelligent Systems," Georgia Institute of Technology, 2004.

[4] Altman, E. Constrained Markov Decision Processes, Chapman & Hall/CRC, 1999.

[5] Anderson, T., "Proactive Use of Vehicle Data in Multi-Model Commercial Operations," Proceedings SAE/NTSB Conference, Alexandria, VA. June 5, 2003.

[6] Bagley D., Youngs R., Mission and Operationally based ZELDA Description, Technical Report, Anteon Corporation, December 2001.

[7] Bajwa, A. and A. Sweet, "The Livingstone Model of a Main Propulsion System," IEEE Aerospace Conference Proceedings, 2003.

[8] Bastos, J.L., Farronato, L.M., Figueroa, H.P., Franzoni, D., Lentijo, S., Monti, A., Smith, A., and Xin Wu, "FPGA-based control of power converter: comparing alternative solutions," Applied Power Electronics Conference and Exposition, Vol. 3, pp. 1983-1989, 6-10 March, 2005.

[9] Bellman, R. "Dynamic Programming". Princeton University Press: Princeton, N. J, 1957.

[10] Browning, T.R., "Applying the Design Structure Matrix to System Decomposition and Integration Problems: A Review and New Directions," IEEE Transactions on Engineering Management, Vol. 48, No. 3, Aug. 2001. pp. 292-306.

[11] Chapra, S.C. and R.P. Canale, Numerical Methods for Engineers, McGraw-Hill, third edition, 1998.

[12] Congressional Budget Data, US House of Representatives Department of Defense Appropriation Bills, 2005.

[13] Cormier, T., Scott, A., Ledsinger, L., McCormick, D., Way, D. and J.R. Olds, "Comparison of Collaborative Optimization to Conventional Design Techniques for a Conceptual RLV," AIAA 2000-4885, 8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Long Beach, CA, September 6-8, 2000.

[14] Davidson, G.L., and T.V. Nguyen, "Human Machine Interface Development Design Guidelines and Cost Estimation," presented at the Thirteenth International Ship Control Systems Symposium (SCSS), Orlando, FL, April 7-9 2003.

[15] D'Epenoux, F. "A Probabilistic Production and Inventory Problem". Management Science. Vol. 10(1): pp.98-108, 1963.

[16] Dieter, G.E., Engineering Design: A Materials and Processing Approach, McGraw Hill, 2000.

[17] Dolgov, D. A. and Durfee, E. H. "Optimal Resource Allocation and Policy Formulation in Loosely-Coupled Markov Decision Processes," in Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling, 2004.

[18] Dunnington L., Stevens H., Grater, G., Integrated Engineering Plant for future naval combatants, Technical Report, Anteon Corporation, January 2003.

[19] Dunnington, L., Garter, G. and H. Stevens, "Integrated Engineering Plant for Future Naval Combatants - Technology Assessment and Demonstration Roadmap," Systems Engineering Group, Engineering Technology Center, Marine Technology Division, Anteon Corporation. 2003.

[20] Edwards, P., "COTS in a Marine Environment – 'Specialist Marine or Industrial P.L.C.' an Insiders Guide for the Future," presented at the Thirteenth International Ship Control Systems Symposium (SCSS) , Orlando, FL, April 7-9 2003.

[21] Ericsen, T., Hingorani N. and Y. Khersonsky, "Power Electronics and Future Marine Electrical Systems," IEEE Transactions on Industry Applications, Vol. 42, No. 1, January/February 2006.

[22] Fairmount Automation. Automated Damage Control System Development, Chilled Water Reduce-Scale Advanced Demonstrator (CW-RSAD), from http://www.fairmountautomation.com/Services/AutomatedDamageControl_RSAD.htm, 2006. [Date Accessed: August, 2006].

[23] Gillis, M.P.W., Keijer, W. and J.L. Meesters, "Risk Based Decision Aid for Damage Control," presented at the Thirteenth International Ship Control Systems Symposium. Orlando Florida, USA. April 7-9 2003.

[24] Gulian, A.M.; Castonguay, L.; Kus, C., Mitchell, C. and G. Ganak, "Shipboard Wireless Network Applications – Solutions to Achieving Desired Metrics," presented at the Thirteenth International Ship Control Systems Symposium (SCSS), Orlando, FL, April 7-9, 2003.

[25] Hazen, G. B. "Stochastic Trees: A New Technique for Temporal Medical Decision Modeling". Medical Decision Making. Vol. 12, 1992.

[26] Hollenberg, J. P. "Markov Cycle Trees: A New Representation for Complex Markov Processes". Medical Decision Making. Vol. 4: 529, 1984.

[27] Howard, R. A. "Dynamic Programming and Markov Processes". MIT Press: Cambridge, MA, 1960.

[28] Hulme, K. F. and C. L. Bloebaum, "A comparison of solution strategies for simulation-based multidisciplinary design optimization," AIAA-1998-4977 AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, 7th, St. Louis, MO, Sept. 2-4, 1998.

[29] IEEE Standards Online. URL: http://standards.ieee.org/reading/ieee/std/lanman/. [Date accessed October 2004]

[30] Integrated Engineering Plant. URL: www.navy.mil [Date accessed July 2005]

[31] Jennings, N.R. and S. Bussmann, "Agent-Based Control Systems-Why Are They Suited to Engineering Complex Systems?", IEEE Control System Magazine, 2003.

[32] Johnson, J. "Biology II: Anatomy & Physiology". http://www.sirinet.net/~jgjohnso/nervous.html. Chapter 50, Part 1. November 2004.

[33] Jung, J. and C.C. Liu, "Multi-Agent Technology for Vulnerability Assessment and Control," Proceedings IEEE PES Summer Meeting, Vancouver, July 2001.

[34] Jung, J., and Liu, C., "Multi-Agent Technology for Vulnerability Assessment and Control", IEEE, 2001

[35] Kam, M., Lebaudy, A., Xu, Y., Lin, E. and M. Wang, "Toward Development of a Virtual Distributed Control System," Report A110324, Department of Electrical and Computer Engineering, Drexel University, Philadelphia, PA. May 2004.

[36] Kirby M., A methodology for Technology Identification Evaluation and Selection in Conceptual and Preliminary Aircraft Design, Ph.D. Thesis, 2001.

[37] Kothare, U.C., and R.K. Rana, "Integrated Machinery Control Systems – Views of a Developing Navy," presented at the Thirteenth International Ship Control Systems Symposium (SCSS), Orlando, FL, April 7-9 2003.

[38] Kroo, I., "Distributed multidisciplinary design and collaborative optimization", VKI lecture series on Optimization Methods & Tools for Multicriteria/Multidisciplinary Design, November 15-19, 2004.

[39] Kroo, I., "Multidisciplinary optimization applications in preliminary design - Status and directions", AIAA-1997-1408, AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference and Exhibit, 38th, and AIAA/ASME/AHS Adaptive Structures Forum, Kissimmee, FL, Apr. 7-10, 1997.

[40] Ledsinger, L. and J.R. Olds, "Multidisciplinary Design Optimization Techniques for Branching Trajectories," AIAA 98-4713, 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, St. Louis, MO, September 2-4, 1998.

[41] Lee, W. H. and W. Scott, Distributed power generation: planning and evaluation, Marcel Dekker, 2000.

[42] Lively, K., Dalessandro, D. and Smith, S. "Complexity Management in Shipboard Automation Architectures Employing Component Level Intelligence," 2003.

[43] Lively, K., Scheidt, D. H. and K. F. Drew, "Mission Based Engineering Plant," ASNE Reconfiguration and Survivability Symposium, Feb 16-18, 2005.

[44] Louden, J., "Total Ownership Cost," NAVSEA Naval Sea Systems Command, 2000.

[45] Louie, R.J., LeBlanc, R.G., and G. Grindstaff, "Digital Video Surveillance System," presented at the Thirteenth International Ship Control Systems Symposium (SCSS), Orlando, FL, April 7-9 2003.

[46] Mandic, D.P. and J.A. Chambers, Recurrent neural networks for prediction: learning algorithms, architectures, and stability, Wiley, New York, 2001.

[47] MathWorks. http://www.mathworks.com/. October 2004.

[48] MATLAB Central: General Communications, URL: http://www.mathworks.nl. [Date accessed October 2004]

[49] Maturana, F., Staron, R., Discenzo, F., Scheidt, D., Pekala, M., Bracy, J., and Michael Zink, "An Interoperable Agent-based Control System for Survivable Shipboard Automation," In Proceedings of ASNE Reconfiguration and Survivability Symposium 2005. February, 2005.

[50] Mavris, D.N., DeLaurentis, D.A., "An Integrated Approach to Military Aircraft Selection and Concept Evaluation," Presented at the 1st AIAA Aircraft Engineering, Technology, and Operations Congress, Anaheim, CA, September 19-21, 1995.

[51] Michalopoulos, P., "Modernization of the Propulsion Plant Monitoring and Control System on the 'S'-Type Frigates. An Approach Using COTSS Components," presented at the Thirteenth International Ship Control Systems Symposium (SCSS), Orlando, FL, April 7-9 2003.

[52] Newell, A. and Simon, H. A. "GPS: A Program That Simulates Human Thought". In Feigenbaum, E. A., and Feldman, J., eds., Computers and Thought: New York, 279--293, McGraw-Hill, 1963.

[53] Ng, H., Guleyupoglu, S., Segaria, F., Malone, S., Woyak, S. and G. Salow, "Collaborative Engineering Enterprise with Integrated Modeling Environment," Paper # 03E-SIW-034, Euro-Simulation Interoperability Workshop, Stockholm, Sweden, June 2003.

[54] OpNet IT-Guru Manual. URL: http://www.opnet.com/products/itguru/ITGuru.pdf. [Date accessed October 2004]

[55] OpNet. URL: www.opnet.com. [Date accessed October 2004]

[56] Parker, E.J., Bettencourt, J.M. "Ships Integrated Information System (SII) – Enhancing Integrated Platform Management System (IPMS). Presented at the Thirteenth International Ship Control Systems Symposium (SCSS) , Orlando, FL, April 7-9 2003.

[57] Rana, R.K. and S. Chhabra, "Control System Architecture for Warship: an Overview," presented at the Thirteenth International Ship Control Systems Symposium (SCSS), Orlando, FL, April 7-9, 2003.

[58] Reliability Assessment Software, RELEX manual. URL: www.relex.com [Date accessed July 2005]

[59] SC-21/ONR S&T Manning Affordability Initiative, ONR and US Navy. URL: http//www.manningaffordability.com. [Date accessed: October 2004]

[60] Scheidt, D., "Intelligent Agent-Based Control," Johns Hopkins APL Technical Digest, Volume 23, Number 4, 2002.

[61] Schwarz, G., "Systems Engineering Study for the Reduced Ships-crew by Virtual Presence(RSVP) Advanced Technology Demonstration (ATD)," May 14, 1999.

[62] Seman, A.J., Tomey, K., and S. Lang, "Reduce's Ship's Crew-By Virtual Presence (RSVP) Advanced Technology Demonstration (ATD)," Report A193314, Naval Surface Warfare Center Carderock Division, Bethesda, MD. February 2001.

[63] Sturtevant, G., Socoloski, P., Bartlett, D. and L. Totimeh, "U.S. Navy Smartship Integrated Ship Controls – Technology Roadmap for Performance Enhancements," presented at the Thirteenth International Ship Control System Symposium. Orlando, Florida, USA. April 7-9 2003.

[64] Sudhoff, S., Żak, S., Zivi, E. and T. McCoy, "Thoughts on Survivability Performance Metrics and an Approach to Integrated Engineering Plant Modeling", White paper, June 2004.

[65] Ulrich, H.G., and M.J. Edwards, "The Next Revolution at Sea," U.S. Naval Institute Proceedings, Vol. 129, No. 10, p. 67, 2003.

[66] Volovoi, V., Kavalieratos, G., Waters, M. and D.N. Mavris, "Modeling the Reliability of Distribution Power Systems using Petri Nets", IEEE, 2004.

[67] Walks, J.P. and J.F. Mearman, "Integrated Engineering Plant," ASNE Reconfiguration and Survivability Symposium, Feb 16-18, 2005.

[68] Wang, Y., Lin, C., "Runge-Kutta neural network for identification of dynamical systems in high accuracy," IEEE Trans. Neural Networks, vol. 9, no. 2, pp294-307, March 1998.

[69] Weston, N.R., Balchanos, M.G., Koepp, M.R., and D.N. Mavris, "Strategies for Integrating Models of Interdependent Subsystems of Complex System-Of-Systems Products," IEEE Proceeding of the Thirty-Eighth Southeastern Symposium on Systems Theory, pages 310-314, March 5, 2006.

[70] Williams, F.W., "DC-ARM Final Demonstration Report," Naval Research Laboratory, Code 6180, June 23, 2003.

[71] Willson, J.M. "ECDIS and International Standards," presented at the Thirteenth International Ship Control Systems Symposium (SCSS), Orlando, FL, April 7-9 2003.

[72] Winkelman, R., "An Educator's Guide to School Networks," URL: http://fcit.coedu.usf.edu/network/ [Date accessed July 2005]

[73] Wooldridge, M., "Reasoning about Rational Agents," Cambridge, MA: MIT Press, 2000.